Title
Neurorobotic Investigation of Biologically Plausible Neural Networks

Author
Zou, Xinyun

Publication Date
2021

Peer reviewed|Thesis/dissertation

UNIVERSITY OF CALIFORNIA,
IRVINE

Neurorobotic Investigation of Biologically Plausible Neural Networks

DISSERTATION

submitted in partial satisfaction of the requirements
for the degree of

DOCTOR OF PHILOSOPHY

in Computer Science

by

Xinyun Zou

Dissertation Committee:
Professor Jeffrey L. Krichmar, Chair
Assistant Professor Emre O. Neftci
Chancellor's Professor Nikil D. Dutt

2021

DEDICATION

To my mom and dad for their unwavering support, understanding and encouragement over the years.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF ALGORITHMS

ACKNOWLEDGMENTS

My dissertation is based upon work supported by Air Force Office of Scientific Research (AFOSR) under Contract No. FA9550-19-1-0306 (Cognitive & Computational Neuroscience), by DARPA under Contract No. FA8750-18-C-0103 (Lifelong Learning Machines: L2M) and Contract No. HR001120C0021 (Science of Artificial Intelligence and Learning for Open-world Novelty: SAIL-ON), and by Toyota Motor North America.

Over the past five years I have received support and encouragement from a great number of individuals. First and foremost I am extremely grateful to my esteemed supervisor, Prof. Jeffrey L. Krichmar, for his invaluable advice, continuous support, and patience during my PhD study. His immense knowledge, plentiful experience, and research enthusiasm have encouraged me in all the time of my academic research and daily life. I also wish to thank my dissertation committee of Prof. Emre Neftci and Prof. Nikil Dutt for their help on my class and research projects as well as practical suggestions on my advancement-to-candidacy and dissertation. I would like to extend my sincere thanks to Prof. Kenneth De Jong, Prof. Douglas Nitz, Dr. Eric 'Siggy' Scott, and Alexander Johnson for their treasured support which was really influential in shaping my experiment methods and critiquing my results in the Air Force Cognitive Maps project. I also very much appreciate Dr. Praveen Pilly for his lead on the DARPA L2M project. Additionally, I must thank all other collaborators worldwide on one or multiple projects. I also had great pleasure of working with my lab mates – Dr. Tiffany Hwu, Kexin Chen, Jinwei Xing, Dr. Hirak Kashyap, Dr. Ting-Shuo Chou, Dr. Emily Rounds, Kenneth Stewart, and Amirhosein Mohaddesi for a cherished time. Last but not least, I would like to express my gratitude to my family and friends, including but not limited to my parents, my grandparents, my sister, and my besties (e.g., Tianyi Zhou, Wen Chen, Yaqi Zhang, Yiqun Amber Zhang, Li Fu, Yun Qu, Zhoutian Shen, Tian Qin, Judith Leng, Xizheng Billy Wan, etc.). Without their tremendous understanding, patience and encouragement over the past years, it would be impossible for me to go through hard times and complete my study.

VITA

Xinyun Zou

EDUCATION

Ph.D. in Computer Science — **June 2021**
University of California, Irvine — *Irvine, CA*

B.S.E. in ECE and BME, Minor in Economics — **May 2016**
Duke University — *Durham, NC*

High School Diploma — **June 2012**
Changzhou Senior High School — *Changzhou, China*

Summer School (Biomedical Ethics, Environ Economics) — **Summer 2010**
Harvard University — *Cambridge, MA*

RESEARCH EXPERIENCE

Graduate Student Researcher — **2016 – 2021**
University of California, Irvine — *Irvine, CA*

Graduate Student Researcher — **Summer 2017**
2017 Telluride Neuromorphic Cognition Workshop — *Telluride, CO*

Undergraduate Research Assistant — **2014 – 2016**
Duke University — *Durham, NC*

TEACHING EXPERIENCE

TA for Discrete Mathematics for Computer Science — **Winter 2020**
University of California, Irvine — *Irvine, CA*

TA for Programming in C/C++ as a Second Language — **Fall2017 – Wtr2018**
University of California, Irvine — *Irvine, CA*

TA for Programming with Python Software Libraries — **Spring 2017**
University of California, Irvine — *Irvine, CA*

TA for Boolean Algebra & Logic — **Winter 2017**
University of California, Irvine — *Irvine, CA*

Reader for Computational Linear Algebra — **Fall 2016**
University of California, Irvine — *Irvine, CA*

REFEREED JOURNAL PUBLICATIONS

1. **Zou, X., Kolouri, S., Pilly, P. K., & Krichmar, J. L. (2020b). Neuromodulated attention and goal-driven perception in uncertain domains.** ***Neural Networks, 125*****, 56-69. doi:10.1016/j.neunet.2020.01.031.**
2. **Kudithipudi, D., ... Clune, J., ... Krichmar, J., ... McNaughton, B., Miikkulainen, R., ... Sejnowski, T., ... Zou, X. (Under Review, 2021). Biological Underpinnings for Lifelong Learning Machines.** ***Nature Machine Intelligence.***
3. Chen, K., Hwu, T., Kashyap, H. J., Krichmar, J. L., Stewart, K., Xing, J., & Zou, X. (2020). Neurorobots as a Means Toward Neuroethology and Explainable AI. *Frontiers in Neurorobotics, 14.* doi:10.3389/fnbot.2020.570308.
4. Krichmar, J. L., Hwu, T., Zou, X., & Hylton, T. (2019). Advantage of prediction and mental imagery for goal-directed behaviour in agents and robots. *Cognitive Computation and Systems, 1*(1), 12-19. doi:10.1049/CCS.2018.0002.

REFEREED CONFERENCE PUBLICATIONS

1. **Zou, X., Scott, E., Johnson, A., Chen, K., Nitz, D., De Jong, K., & Krichmar, J. (2021, July).** ***Neuroevolution of a Recurrent Neural Network for Spatial and Working Memory in a Simulated Robotic Environment.*** **Peer-reviewed poster accepted in Proceedings of 2021 Genetic and Evolutionary Computation Conference Companion (GECCO '21 Companion).** (arXiv preprint for full paper available at arXiv:2102.12638. Also in preparation for journal submission with advanced cognitive map results and analysis.)
2. **Zou, X., Hwu, T., Krichmar, J., & Neftci, E. (2020a). Terrain Classification with a Reservoir-Based Network of Spiking Neurons.** ***Proceedings of 2020 IEEE International Symposium on Circuits and Systems (ISCAS 2020)*** **(pp. 1-5). doi:10.1109/ISCAS45731.2020.9180740.**
3. **Hwu, T., Krichmar, J.L., & Zou, X. (2017). A Complete Neuromorphic Solution to Outdoor Navigation and Path Planning.** ***Proceedings of 2017 IEEE International Symposium on Circuits and Systems (ISCAS 2017)*** **(pp. 2707-2710). doi:10.1109/ISCAS.2017.8050981.**
4. Chen, K., Johnson, A., Scott, E.O., Zou, X., De Jong, K.A., Nitz, D.A., & Krichmar, J. (2021, July). *Differential Spatial Representations in Hippocampal CA1 and Subiculum Emerged in Evolved Spiking Neural Networks.* Paper accepted in Proceedings of 2021 IEEE International Joint Conference on Neural Networks (IJCNN 2021).
5. Xing, J., Nagata, T., Chen, K., Zou, X., Neftci, E., & Krichmar, J. (2021, February). *Domain Adaptation In Reinforcement Learning Via Latent Unified State Representation.* Paper presented at the 35th AAAI Conference on Artificial Intelligence (AAAI-21). (arXiv preprint available at arXiv:2102.05714.)
6. Xing, J., Zou, X., & Krichmar, J. L. (2020). Neuromodulated Patience for Robot and Self-Driving Vehicle Navigation. *Proceedings of 2020 IEEE International Joint Conference on Neural Networks (IJCNN 2020)* (pp. 1-8). doi:10.1109/IJCNN48605.2020.9206642.

REFEREED PREPRINT

1. Kolouri, S., Ketz, N., Zou, X., Krichmar, J., & Pilly, P. (2019). Attention-Based Structural-Plasticity. *arXiv preprint arXiv:1903.06070*.

SPECIAL PRESENTATIONS

Invited Talk: Attention and Navigation Strategies for Neurorobotics and Neuromorphic Applications — **June 2019**
RSS 2019 Workshop: Pervasively Neural-Dynamic Robotics — Freiburg, Germany

Spotlight Talk: Terrain Classification With A Reservoir-Based Network of Spiking Neurons — **April 2019**
2019 Southern California Robotics Symposium (SCR) — Caltech, Pasadena, CA

Poster: Neuromodulated Goal-Driven Perception in Uncertain Domains — **June 2019**
2019 Joint Symposium on Neural Computation (JSNC) — USC, Los Angeles, CA

TECHNICAL SKILLS

Languages: Python, C/C++, MATLAB, Java, C#
Libraries: PyTorch, TensorFlow, Keras, OpenCV
Developer Tools: Webots, CARLA, Android Studio, ROS, Docker, CHASE-CI Cloud, Visual Studio, Arduino, Simulink, Logisim, SOLIDWORKS, AnimatLab, CARLsim, Brian2, DYNAP-SE
Operating Systems: Linux, Windows

ACTIVITIES & SOCIETIES

2nd Place for Multiple Ground Robots at 2017 UCI Search and Rescue Robotics Invitational — **May 2017**
Student Advisor for UCI Undergraduate Ground Robotics Team — Irvine, CA

Top 10 (out of 250 teams) and Best Team for Startup seen by 1517 Fund (1517 Prize) at LA Hacks 2017 — **April 2017**
Robotics Developer in CARLsitter Team — Los Angeles, CA

3rd Place for Battery-Electric Prototype at 2013 Shell Eco-Marathon Americas — **April 2013**
Car Builder and Racer in Duke Electric Vehicles Team — Durham, NC

ABSTRACT OF THE DISSERTATION

Neurorobotic Investigation of Biologically Plausible Neural Networks

By

Xinyun Zou

Doctor of Philosophy in Computer Science

University of California, Irvine, 2021

Professor Jeffrey L. Krichmar, Chair

My dissertation focuses on three research problems to investigate how the robot's behavior leads to a qualitative and quantitative explanation of neural activities, and vice versa, that is, how neural activities lead to behavior. In the first problem, we simulated a rat in a robot simulator to replicate the behavior and neural activity observed in rats during a spatial and working memory task. A recurrent neural network (RNN) with sensory and vision inputs was evolved to control the robot motor wheels and navigate a virtual T-maze. Our current findings suggest that neurons in the RNN are performing mixed selectivity and conjunctive coding. Moreover, the RNN activity resembles spatial information and trajectory-dependent coding observed in the hippocampus (Zou et al., 2021). In the second problem, we developed a goal-driven perception algorithm inspired by effects of the cholinergic (ACh) and noradrenergic (NE) neuromodulatory systems on attention and tracking uncertainties. We tested the network architecture, which extended the contrastive excitation backprop (c-EB), in a noisy MNIST-pair task and an action-based human support robot task. The network architecture could quickly learn the context without supervision, flexibly apply attention to the appropriate goal, and rapidly detect and re-adapt to context changes (Zou et al., 2020b). In the third problem, we developed a reservoir-based spiking neural network (r-SNN) to classify three terrain types in a botanical garden. The input spike trains were generated from the linear accelerometer, gyroscope, and image data collected by a six-wheel Android-based

robot (ABR). Our r-SNN terrain prediction can be used to evaluate the cost of traversal for path planning. It is a promising approach to develop a complete neuromorphic robot navigation system capable of operating over long durations with minimal power consumption (Zou et al., 2020a). We suggest that neurorobotic investigation of biologically plausible neural networks can be a powerful methodology for understanding neuroscience, as well as for artificial intelligence and machine learning.

Chapter 1

Introduction

Biological intelligence incorporates both conscious and subconscious knowledge of a human being or an animal. It is sophisticated with a huge space left for people to further explore. However, existing biological data and theories can already inspire people to build artificially intelligent systems. For a higher chance of survival, different aspects of biological intelligence powerfully work together to absorb and filter all kinds of information at the same time. Meanwhile, it is also energy efficient. For instance, the power consumption of a human adult is only 20 Watts (Sokoloff, 1960).

While there are many interesting and important studies on artificial motor control systems which mimic animals' locomotion (Lock et al., 2013; Ijspeert, 2014), my doctoral research instead focuses on adapting the brain signals and operating mechanisms on real-world applications via neurorobotic implementations. My studies may also benefit in the opposite direction, that is, to provide possible explanations of some aspects of biological intelligence. Compared with naturally evolved animals and human beings, most existing bio-inspired systems have much reduced complexity to utilize certain biological concepts for targeted functionalities and thus become easier to experiment with. Figure 1.1 illustrates the sum-

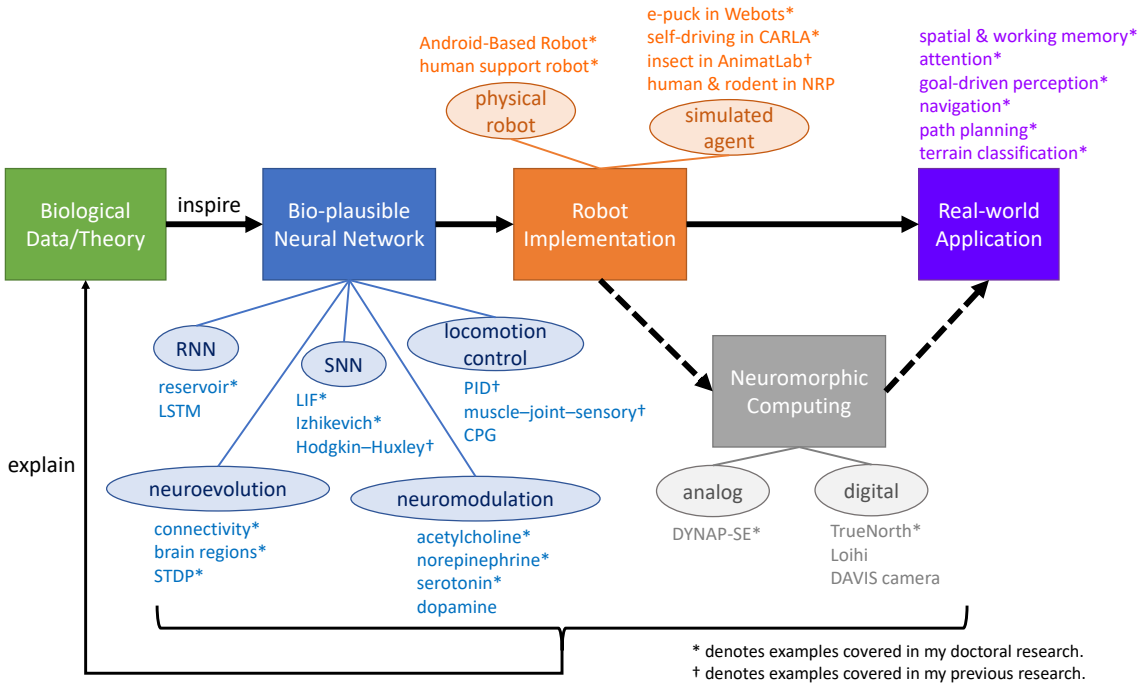


Figure 1.1: Xinyun Zou's research summary chart. The research projects all investigate how neural activities lead to the robot's behavior, and vice versa, that is, how behavior leads to a qualitative and quantitative explanation of neural activities.

mary chart for connections among my research topics.

For the rest of this chapter, I will introduce the background for neurorobotics and biologically plausible neural networks, as well as mechanisms which are utilized in my doctoral research projects. Then my dissertation will focus on three research problems to investigate how the robot's behavior leads to a qualitative and quantitative explanation of neural activities, and vice versa, that is, how neural activities lead to behavior. In the end of my dissertation, I will talk about potential future directions based on my existing studies and research interests.

1.1 Background

1.1.1 Neurorobotics

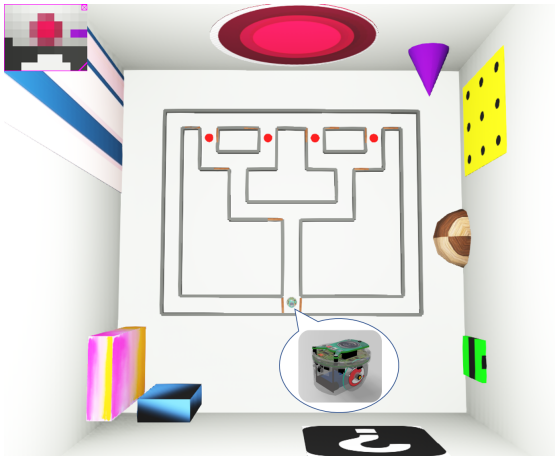
Neurorobots are robots whose control has been modeled after some aspect of the brain. Since the brain is so closely coupled to the body and situated in the environment, neurorobots can be a powerful tool for studying neural function in a holistic fashion (Krichmar, 2018). In a neurorobot experiment, the robot operates in the real world. It takes noisy sensory information from its environment and integrates this into actions. While this behavior is occurring, the neurorobotic researcher has the ability to examine the complete brain, that is, every neuron and synaptic change. Similar to a neuroethologist, but with far more control, the neuroroboticist can explain how these artificial brains give rise to behavior (Chen et al., 2020). Figure 1.2 shows the physical and simulated robot implementations covered in my doctoral research.

Physical Robots

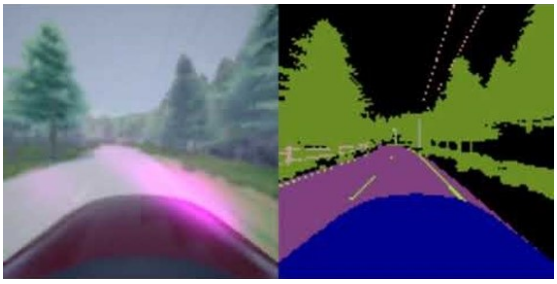
The Android-Based Robotics (ABR) Platform designed in our UCI CARL lab (Oros and Krichmar, 2013) (shown in Figure 1.2a) can accomplish various outdoor navigational tasks including path planning, terrain classification and road following (Hwu et al., 2017b; Zou et al., 2020a; Hwu et al., 2017a). This robot runs on a Dagu Wild Thumper 6-wheel-drive all-terrain chassis, with an SPT 200 pan and tilt to hold the Samsung Galaxy S5 smartphone and control the view of the phone camera. Front-facing MaxBotix LV-MaxSonars can detect obstacles. An ION Motion motor controller and IOIO-OTG microcontroller are housed in the back of the robot. Computing is handled by the Android phone, which accesses the sensors and actuators through a Bluetooth connection with the IOIO-OTG (Hwu et al., 2017a).

(a) Outdoor Android Based Robot (ABR)

(b) Indoor Toyota Human Support Robot (HSR)



(c) E-puck in the Webots simulator



(d) Self-driving in the CARLA simulator

Figure 1.2: Physical and simulated robot implementations covered in Xinyun Zou's doctoral research. (a–c) Images of an Android-Based Robot (ABR), the Toyota Human Support Robot (HSR), and a simulated e-puck in their corresponding testing environments are adapted from our original papers (Zou et al., 2020a,b, 2021) with permission. (d) A self-driving testing scenario within the CARLA simulator (Dosovitskiy et al., 2017) and its semantic segmentation output are for one of our ongoing projects.

The Human Support Robot (HSR) developed by Toyota (Yamamoto et al., 2018) (shown in Figure 1.2b) is mainly used indoor to support human daily life (Zou et al., 2020b; Hwu et al., 2020). It has 8 DoF to support flexible movement of the mobile base, arm, and torso lift. Various sensors (e.g., laser range sensor, bumper sensor, head/hand stereo and wide cameras, etc.) help with accurate visual processing, obstacle avoidance, and simultaneous localization and mapping (SLAM) (Yamamoto et al., 2018; Bailey and Durrant-Whyte, 2006). In a schema recognition experment, by learning schemas in the form of objects belonging to different rooms, the Toyota HSR robot can disambiguate task commands, such as using its current context to pick up a book (Hwu et al., 2020). In a neuromodulated goal-driven perception experiment, the HSR's attention is allocated to the desired action/object pair (the cholinergic system) and adjust to the change of goals in an uncertain domain (the noradrenergic system) (Zou et al., 2020b).

Robotic Simulators

Some of our projects are computationally expensive or relatively more difficult to directly apply on a physical robots. In such cases, some robotic simulators can provide efficient system control and signal processing without loss of much accuracy or realism in the simulated robot design.

The Webots simulator (Michel, 2004) currently includes more than 20 types of robot models, ranging from multi-wheeled robots (e.g., E-puck, Pioneer 3-AT, Sojourner, etc.) to multi-ped ones (e.g., Atlas, Spot, Mantis, etc.). Users can also program the properties of a large set of sensors and actuators (e.g. proximity sensors, accelerometers, cameras, lidars, GPS, emitters and receivers, LEDs, grippers, IMU, etc.) as well as self define the 3D testing "world" with different objects (Dosovitskiy et al., 2017). Aside from a few class projects, I mainly used Webots in our cognitive map project. Figure 1.2c shows our simulated e-puck running in a triple T-maze built from scratch. We also programmed two controllers, one to supervise

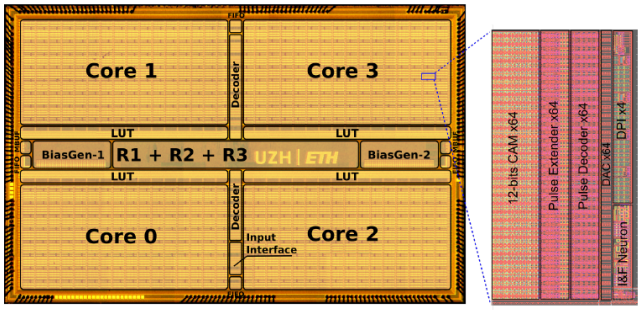
the task of maze navigation and evaluate each genotype's performance in each generation whereas the other to actuate the robot motor via the recurrent network with sensor inputs.

The CARLA simulator (Dosovitskiy et al., 2017) targets self-driving research with the car in various urban layouts. In this platform, users can specify sensor suites, control all static and dynamic actuators, generate maps, define different traffic situations, and add other digital assets (e.g., other vehicles, pedestrians, and buildings) into the testing environment (Dosovitskiy et al., 2017). We are currently using CARLA to work on an architecture for neuromodulated attention and task-driven perception with CARLA within the Scorpius software framework for a reinforcement learning scenario. As illustrated in Figure 1.2d, the self-driving car would pay attention to different regions of its front view based on a perturbation-based saliency map for different tasks (e.g., aggressive driving v.s. passive driving).

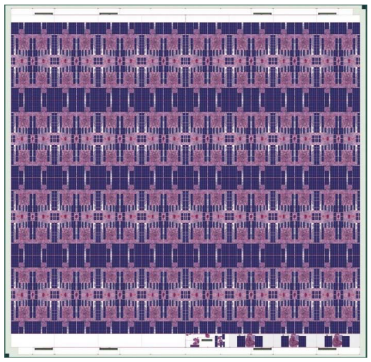
1.1.2 Neuromorphic Computing

Neuromorphic computing is a concept developed by Carver Mead in the late 1980s (Mead, 1990). The original design utilizes very-large-scale integration (VLSI) systems with electronic analog circuits to represent biological networks in the real nervous system. Nowadays, the term "neuromorphic" covers a much broader range of designs including analog, digital, mixed-mode analog/digital VLSI, as well as algorithms that implement biologically plausible spiking neural networks (Greengard, 2020).

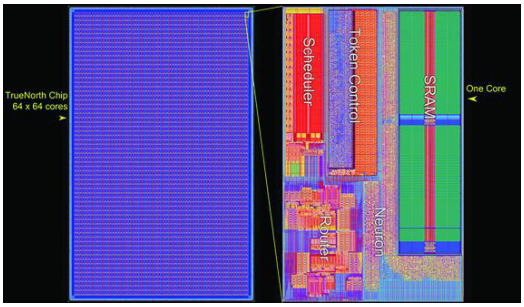
An analog neuromorphic hardware, such as the DYNAP-SE (Moradi et al., 2017), usually achieves better accuracy in signal processing but is also more difficult to tune, leading to more experimental errors sometimes (see Figure 1.3a). To further scale up the computing and reduce noises, digital neuromorphic chips have become more and more popular (see Figure 1.3b–1.3c), such as Intel Loihi (Davies et al., 2018), IBM TrueNorth (Akopyan et al.,



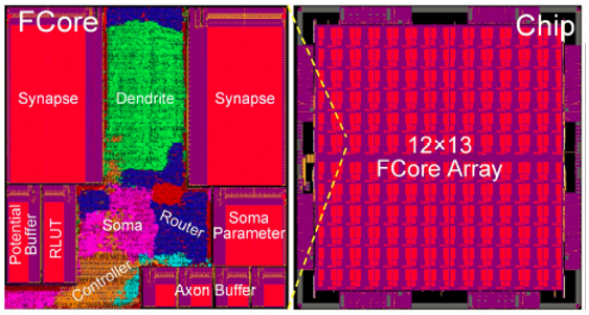
(a) Analog DYNAP-SE Chip (SNN)



(b) Digital Intel Loihi Chip (SNN)



(c) Digital IBM TrueNorth Chip (SNN)



(d) Digital Tianjic Chip (hybrid SNN and ANN)

Figure 1.3: Neuromorphic chip layouts for (a) DYNAP-SE, (b) Intel Loihi, (c) IBM TrueNorth, and (d) Tianjic. Images are adapted from (Moradi et al., 2017; Davies et al., 2018; Akopyan et al., 2015; Pei et al., 2019).

2015), and the DAVIS camera (Yang et al., 2015). There is also the hybrid Tianjic chip architecture (Pei et al., 2019) which takes the advantages of both the neuroscience-oriented spiking neural network (SNN) and the artificial neural network (ANN) (see Figure 1.3d).

Although it is not an necessary component, adding neuromorphic control to a compatible neurorobotic system can significantly reduce energy cost with event-driven, parallel computing (Hwu et al., 2017b,a).

1.1.3 Spiking Neural Networks (SNNs)

Spiking neural networks (SNNs) are artificial neural networks that more closely mimic natural neural networks. Each neuron in the SNN fires and propagates information to a postsynaptic

neuron only when its membrane potential reaches a pre-defined threshold (Gerstner and Kistler, 2002; Dayan and Abbott, 2001). SNNs can take advantage of the neuromorphic hardware, because each neuron computes its state independently, making the SNN parallel, and spikes are asynchronous events (Zou et al., 2020a; Hwu et al., 2017b).

Leaky Integrate-and-Fire Model

One commonly used spiking neuron model is the leaky integrate-and-fire (LIF) model. The major advantage is its computational efficiency, because it retains the minimal ingredients of membrane dynamics (Gerstner and Kistler, 2002). In this model, for each postsynaptic neuron i at each time step t, if it is not within the refractory period, the postsynaptic membrane potential (U_i) will be updated via the differential equation

$$\frac{dU_i}{dt} = \frac{U^{rest} - U_i}{\tau^{mem}} + I_i^{syn}(t), \tag{1.1}$$

where U^{rest} is the resting membrane potential, τ^{mem} is the membrane time constant, and $I_i^{syn}(t)$ is the synaptic input current. $I_i^{syn}(t)$ jumps by summation of the weight w_{ij} upon spike arrival from each presynaptic neuron j (i.e., when $S_j(t) = 1$), with the equation shown below

$$\frac{d}{dt} I_i^{syn}(t) = -\frac{I_i^{syn}(t)}{\tau^{syn}} + \sum_{j \in pre} w_{ij} S_j(t). \tag{1.2}$$

When U_i reaches the threshold θ^{mem} and the neuron i is not in the refractory period, a spike is triggered (i.e., $S_i(t) = 1$). The neuron then remains refractory for n^{ref} time steps (Zou et al., 2020a).

Hodgkin–Huxley Model

A much more biologically plausible spiking neuron model is the Hodgkin–Huxley (H&H) model (Hodgkin and Huxley, 1952c). It links the flow of ionic currents across the neuronal cell membrane with its membrane potential. Hodgkin and Huxley computed a set of nonlinear differential equations to describe the behavior of ion channels (i.e., sodium, potassium, and a leak one with mainly chloride ions) that permeate the cell membrane of the squid giant axon. Each ion channel is characterized with a different conductance, whereas the semipermeable cell membrane acts as a capacitor (Hodgkin and Huxley, 1952c; Hodgkin et al., 1952; Hodgkin and Huxley, 1952a,b). This model is computationally prohibitive and can only simulate a few neurons in real-time.

Izhikevich Model

There is another model, called the Izhikevich model (Izhikevich, 2003), which finds a balance between biological plausibility and computational efficiency. Unlike the Hodgkin-Huxley model, the Izhikevich model does not account for the biophysics of neurons. It uses mathematical equations to compute a wide range of spiking patterns for cortical neurons. Therefore, this model is both biologically realistic and capable of simulating large-scale spiking neurons in real-time (Izhikevich, 2004).

1.1.4 Recurrent Neural Networks (RNNs)

A recurrent neural network (RNN) uses connections between internal neurons to form a directed cycle. Its internal state serves as the memory to process arbitrary sequences of inputs. RNNs have been used for a variety of applications, such as spatial navigation, terrain classification, motion prediction, health monitoring, speech recognition, and time

series forecasting (Zou et al., 2021, 2020a; Kashyap et al., 2018; Das et al., 2018; Graves et al., 2013; Hewamalage et al., 2019). In our cognitive map project, we designed an RNN system to replicate the behavior and neural activity observed in rats for a triple T-maze experiment. The rat was simulated in the Webots robot simulator and used vision, distance and accelerometer sensors to navigate a virtual maze with rewards. The RNN activity resembles spatial information and trajectory-dependent coding observed in the hippocampus (Zou et al., 2021).

Reservoir Computing

The recurrence can be tractably harnessed using a reservoir-based approach, such as the liquid state machine (LSM) (Maass et al., 2002) and the echo state network (ESN) (Jaeger, 2007). Reservoir computing is an RNN framework that maps input signals into higher dimensional computational spaces through the dynamics of a fixed, non-linear system called a reservoir (Schrauwen et al., 2007). The reservoir dynamics is fixed after the recurrent weights are initialized randomly. After the input signal is fed into the reservoir, only the readout is trained to read the state of the reservoir and map it to the desired output (Tanaka et al., 2019). This technique is utilized in our terrain classification project, with input from the spike trains of the robot's sensor signals and output to guide prediction of three terrain types via supervised learning (Zou et al., 2020a).

Long Short-Term Memory Units

Long short-term memory units (LSTMs) (Hochreiter and Schmidhuber, 1997) extend the memory in recurrent neural networks so that the system can learn from important experiences over a long period of time. The LSTM can read, write, and delete information from its memory. This memory can be considered as a gated cell which decides whether to store

or delete information based on its assigned importance learned over time. That is, the cells decide when to open or close gates through the iterative process of guessing, error backpropagation, and weight adjustment via gradient descent (Hochreiter and Schmidhuber, 1997).

1.1.5 Neuromodulation

Neuromodulatory systems in the brain can have a strong contextual effect on large swaths of downstream brain areas. As illustrated in Figure 1.4, these neurons release neurotransmitters that have both a local effect and a global effect on activity and plasticity. The neuromodulators adapted in our projects include acetylcholine (ACh), norepinephrine (NE), serotonin (5-HT), and dopamine (DA) (Krichmar, 2008). ACh regulates the trade-off between stimulus-driven and goal-driven attention (Zou et al., 2020b). NE drives responses to novelty and surprises (Dayan and Yu, 2006). 5-HT can shift patience and assertiveness depending on the context (Miyazaki et al., 2018). DA shifts neurons allows for associating cues with predicting outcomes, which can be rewards, punishment, and novelty (Wise, 2004). All these neuromodulators have a selective effect on learning.

Neuromodulation has been studied and modeled in the context of its role in behavioral adaptation in the presence of expected and unexpected uncertainties (Dayan and Yu, 2006). Successful autonomous lifelong learning agents, no matter if they are biological or artificial, must possess internal mechanisms that allow them to monitor and gauge performance against expectations.

The influence of the ACh system and NE systems on goal-directed perception was studied in an action-based attention task using the Toyota HSR (Zou et al., 2020b). In our experiment, a robot was required to attend to goal-related objects (the ACh system) and adjust to the change of goals in an uncertain domain (the NE system). Four different actions (i.e., “eat”,

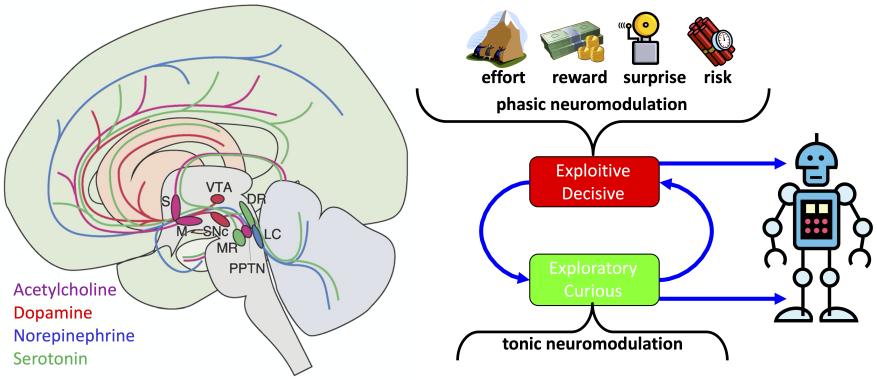


Figure 1.4: Neuromodulatory systems in the brain. Left. The source of neuromodulators are subcortical. Acetylcholine originates in the Substantia Innominata (S) and in the Medial Septum (M). Dopamine originates in the Ventral Tegmental area and the Substantia Nigra Compacta (SNc), Norepinephrine originates in the locus coeruleus (LC), and Serotonin originates in the dorsal raphe (DR) and medial raphe (MR) nuclei. These sources project to large areas of the nervous system. Figure adapted from (Doya, 2002). Right. Phasic neuromodulation drives the agent toward more exploitive and decisive behavior, and tonic neuromodulation drives the agent toward more exploratory or curious behavior. The activity of each neuromodulator is related to environmental stimuli. For example, acetylcholine levels appear to be related to attentional effort, dopamine levels appear to be related to reward anticipation, norepinephrine levels appear to be related to surprise or novelty, and serotonin levels appear to be related to risk assessment and impulsiveness. Adapted from (Krichmar, 2008).

"work-on-computer", "read", and "say-hi") were available in the experiment and each of them was associated with different images of objects. For example, the goal action "eat" might result in attention to objects such as "apple" or "banana" while the action "say-hi" should lead attention to a "person". During the experiment, the goal action changed periodically and the robot needed to select the action and object it thought the user wanted based on prior experience. Our model demonstrated how neuromodulatory systems can facilitate rapid adaptation to change in uncertain environments. The goal-directed perception was realized through the allocation of the robot's attention to the desired action/object pair (Zou et al., 2020b; Dayan and Yu, 2006; Chen et al., 2020).

5-HT activity is thought to be important for regulating anxious behavior and harm aversion. But recently, 5-HT has been shown to have an influence on patience control (Miyazaki et al., 2018). To test this idea in a real-world application, (Xing et al., 2020) designed a robotic navigation experiment to show how changing the simulated 5-HT level could affect the amount of time the robot spent searching for a desired location. In our experiment, the robot searched for GPS waypoints in different outdoor environments. If the 5-HT level was low or a waypoint was difficult to find, the robot became impatient and searched for another waypoint. From this, flexible navigation strategies emerged in the observed robot behavior, such as calling off the search of a difficult to find landmark due to impatience or taking advantage of a smoother but longer route by being extra patient (Xing et al., 2020; Chen et al., 2020).

1.1.6 Neuroevolution

To survive and develop well in this world, each organism has undergone a long process of evolution over many generations and even split to form different species. Evolutionary robotics is a method for building control system components or the morphology of a robot (Bongard,

2013; Nolfi et al., 2016). The biological inspiration behind this field is Darwin's theory of evolution, which was constructed with three principles. (1) *Natural Selection*: genotypes that can be well adapted to their environments are more likely to survive and reproduce. (2) *Heredity*: the fittest genotypes from the previous generation can be directly kept in the next generation; moreover, the new offspring in each generation is generated based on the selected genes from two parents. (3) *Variation*: the new offspring goes through mutations and crossover with a certain probability and thus differs from both parents (Winther, 2000; Gayon, 1998).

Main Classes of Evolutionary Algorithms (EA)

Evolutionary computation supports the belief that complex structure and behavior can be generated from the combination of selection, inheritance, and random variations. The four main EA types are evolutionary strategies (ES), evolutionary programming (EP), genetic algorithms (GA), and genetic programming (GP). They are different in their selection strategies, primary representations, and balance between mutation and crossover (Downing, 2015).

ES was invented by Rechenberg and Schwefel in the 1970s (Rechenberg, 1973; Schwefel, 1977; Schwefel and Rudolph, 1995). Individuals are represented as vectors of real numbers. It traditionally uses the overproduction of children followed by either full generational replacement or more commonly generational mixing (i.e., by competing with their parents) (Fogel, 1997; Hansen, 2006). Since its mutation operators are encoded in the individual genome under evolutionary control, good strategy parameters pass down to the next generation by selecting the fittest individuals (Downing, 2015).

Different from the other EAs, EP represents an entire species with each phenotype. There is no mating between different species in the EP, so it has no genotype recombination (i.e., no crossover), which indicates that each species needs to generate offspring. Individual

genomes are coded as integer vectors that include states, transitions, and output conditions for a finite-state automaton (FSA) (Fogel et al., 1966; Fogel and Corne, 2002).

GA has a bit-string representation. It usually applies generational replacement to have identical offspring and parent population sizes, but sometimes applies generational gap as an alternative to have a smaller offspring population size (Holland, 1992). Therefore, mating selection instead of mutation is the main emphasis in the GA, whereas the "EP forbids crossover and ES uses it sparingly" (Downing, 2015). Deep GAs such as NEAT and its extensions will be introduced below.

GP was developed in 1990's by John Koza (Koza and Koza, 1992; Koza, 1994; Koza et al., 2006). The major characteristics of the GP is that it evolves programs instead of parameter lists. It applies evolutionary search to the space of tree structures (i.e., non-linear chromosomes). The average tree sizes tend to increase over time, despite that it has limitation on the maximum tree size and penalty for being oversized. Mutation represented by random changes in the trees is possible but not necessary. Recombination in the GP means to exchange subtrees (Downing, 2015).

In recent times, concepts from different EA types may be integrated, so some people may simply call their method as an "evolutionary algorithm" (Scott and Luke, 2019; Zou et al., 2021; Charvet et al., 2011). Figure 1.5 shows a general procedure for neuroevolution.

NEAT v.s. HyperNEAT

Another popular evolutionary mechanism is NEAT, which has its unique feature of evolving the network topology together with the weights (Stanley and Miikkulainen, 2002). HyperNEAT extends NEAT by evolving connective CPPNs that generate patterns with regularities (e.g., symmetry, repetition, repetition with variation, etc.) (Stanley et al., 2009). In the case of quadruped locomotion investigated by Clune et al. (2009), HyperNEAT could evolve

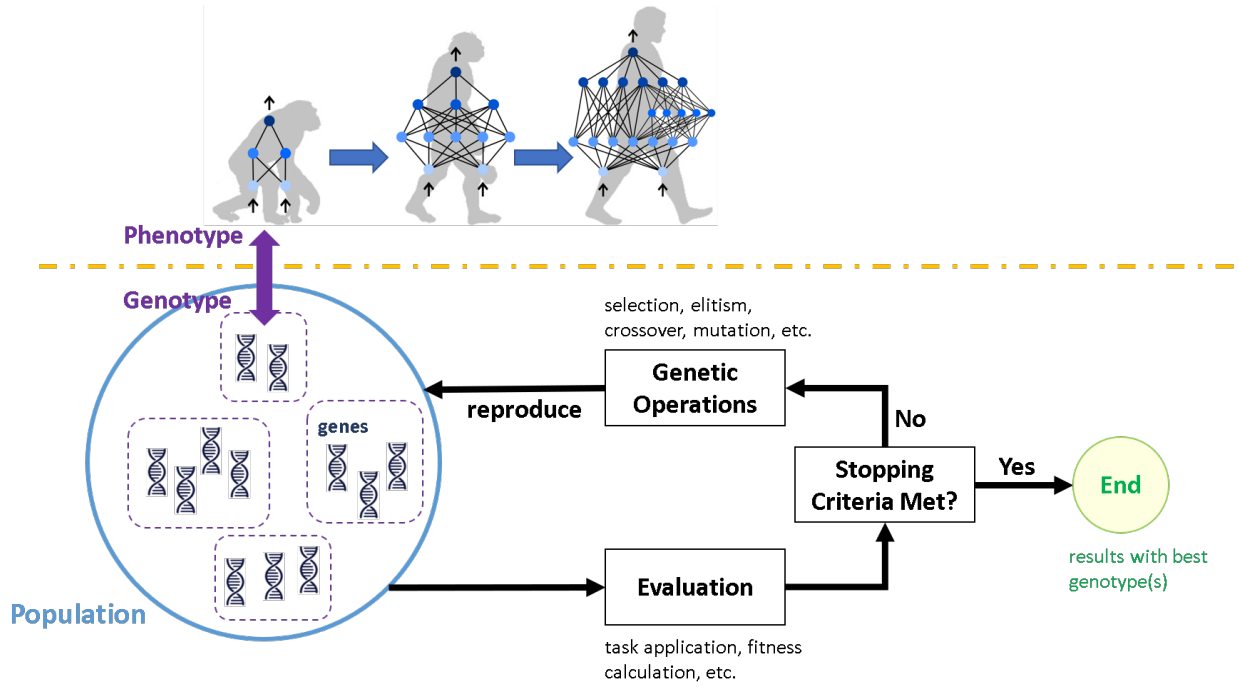


Figure 1.5: General procedure for neuroevolution. The upper subplot is adapted from (Pauls, 2020).

common gaits by exploiting the geometry to generate front-back, left-right, or diagonal symmetries. Our current model was tuned to have a fixed number of recurrent neurons since the first generation and have all-to-all connections between two layers or within the recurrent layer. It may be of interest to combine NEAT/HyperNEAT with topologies of RNNs to solve more complex problems and scenarios. For example, NEAT might be utilized in the beginning of the evolutionary process to efficiently derive a morphology for a more standard evolutionary algorithm to use in later generations. This hybrid approach has similarities to Akinci and Philippides (2019).

Utilization of Evolutionary Algorithms

We have used evolutionary algorithms in our cognitive map projects. In one of them, we evolved weights in a biologically plausible recurrent neural network (RNN) to replicate the behavior and neural activity observed in rats for a triple T-maze experiment. Our evolved

RNN encoded not only spatial information but also working memory to remember which paths had been traversed recently and which paths remained to be explored (Zou et al., 2021). In a related project led by my colleague Kexin Chen (accepted in IJCNN 2021), the modeling of CA1 and SUB with spiking neural networks (SNNs) was optimized by evolving STDP-H parameters. The resulting networks show highly place-specific responses in CA1 neurons and the emergence of pattern recurrence in the spatially specific firing of SUB neurons.

1.1.7 Neurons for Hippocampal Formation

The hippocampal formation is crucial for spatial memory and navigation (Andersen et al., 2007; Ferbinteanu and Shapiro, 2003; Olson et al., 2017). There are several kinds of neurons within the hippocampus or as inputs to the hippocampus that may be associated with cognitive maps (Tolman, 1948), including place cells, head-direction cells, and grid cells (Zou et al., 2021; Banino et al., 2018).

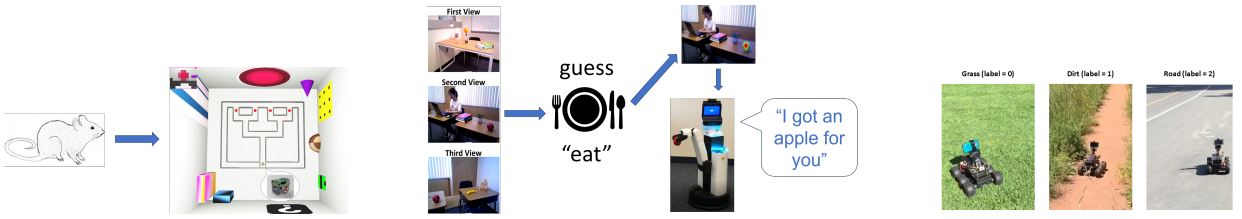
Hippocampal "place cells" are activated selectively when an animal enters their "place fields", which can signal the allocentric position of the animal during a navigational task (O'keefe and Nadel, 1978; Wilson and McNaughton, 1993b; Moser et al., 2008). The place cells are observed in both CA1 and subiculum (SUB) regions. Compared to those in CA1, SUB place cells showed larger and less specific place fields (Potvin et al., 2007), and exhibit more directional modulation for field activities (Sharp and Green, 1994; Olson et al., 2017). The neural compass is implemented by "head-direction cells" (Taube et al., 1990; Wiener and Taube, 2005), which encode the directionality of the animal's head regardless of the actual location. They are found in multiple brain regions, such as SUB, retrosplenial cortex, and entorhinal cortex (Taube, 2007; Chen et al., 1994; Giocomo et al., 2014). A more abstract representation of combined knowledge for location, direction, distance, and speed is seen

by "grid cells". They are mainly in the entorhinal cortex, which fire in a repeating pattern when the animal reaches fields that form a hexagonal lattice across the environment (Hafting et al., 2005; Barry et al., 2007).

The neural activities appeared in hippocampal formation have been adapted or replicated in some artificial networks. For example, we evolved a RNN to control a simulated robot in a spatial and working memory task (Zou et al., 2021). At the population level, the evolved RNN activity has similar characteristics as the hippocampus. Similar to CA1, the RNN received speed, direction, and visual information as input, and combined these types of sensory information to construct a journey-dependent place code (Taube et al., 1990; Sargolini et al., 2006; Kropff et al., 2015; O'Keefe, 1976; Hafting et al., 2005; Sun et al., 2019; Potvin et al., 2007; Frost et al., 2020). In a related project led by my colleague Kexin Chen (accepted in IJCNN 2021), optimized by evolving STDP-H parameters and compared with biological experimental data collected from rats, our simulated networks of CA1 and SUB with spiking neural networks (SNNs) show highly place-specific responses in CA1 neurons and the emergence of pattern recurrence in the spatially specific firing of SUB neurons. In a well-known project done by Banino et al. (2018), an LSTM was used to emerge grid-cell representations for path integration with simulated trajectories for foraging rodents. The output of this LSTM projected to place and head direction units via a linear layer for regularization. Their findings supported that grid cells are critical for vector-based navigation and can be combined with other path planning strategies for complex spatial tasks.

1.2 Overview of Main Projects

For my doctoral research, I conducted three main projects that cover most of the concepts described in Section 1.1. The first project is related to the cognitive map concept (see Figure



(a) Cognitive Map (b) Goal-Driven Perception (c) Terrain Classification

Figure 1.6: Three main projects in Xinyun Zou's doctoral research. Images are adapted from (Zou et al., 2021, 2020b,a).

1.6a). The second is based on neuromodulated goal-driven perception (see Figure 1.6b). The third links to terrain classification (see Figure 1.6c). Their abstracts are introduced below in this section. Chapters 2–4 include their detailed experimental setups, results, and discussions.

1.2.1 Neuroevolution for Spatial and Working Memory in a Maze

(This subsection is reprinted, with permission, from the abstract of a previous preprint (Zou et al., 2021).)

Animals ranging from rats to humans can demonstrate cognitive map capabilities. We evolved weights in a biologically plausible recurrent neural network (RNN) using an evolutionary algorithm to replicate the behavior and neural activity observed in rats during a spatial and working memory task in a triple T-maze. The rat was simulated in the Webots robot simulator and used vision, distance and accelerometer sensors to navigate a virtual maze. After evolving weights from sensory inputs to the RNN, within the RNN, and from the RNN to the robot's motors, the Webots agent successfully navigated the space to reach all four reward arms with minimal repeats before time-out. Our current findings suggest that it is the RNN dynamics that are key to performance, and that performance is not dependent on any one sensory type, which suggests that neurons in the RNN are performing mixed selectivity and conjunctive coding. Moreover, the RNN activity resembles spatial in-

formation and trajectory-dependent coding observed in the hippocampus. Collectively, the evolved RNN exhibits navigation skills, spatial memory, and working memory. Our method demonstrates how the dynamic activity in evolved RNNs can capture interesting and complex cognitive behavior and may be used to create RNN controllers for robotic applications. See Chapter 2 for details.

1.2.2 Neuromodulated Attention and Goal-Driven Perception

In uncertain domains, the goals are often unknown and need to be predicted by the organism or system. In this project, contrastive Excitation Backprop (c-EB) was used in two goal-driven perception tasks – one with pairs of noisy MNIST digits and the other with a robot in an action-based attention scenario. The first task included attending to even, odd, low, and high digits, whereas the second task included action goals, such as "eat", "work-on-computer", "read", and "say-hi" that led to attention to objects associated with those actions. The system had to increase attention to target items and decrease attention to distractor items and background noise. Because the valid goal was unknown, an online learning model based on the cholinergic and noradrenergic neuromodulatory systems was used to predict a noisy goal (*expected uncertainty*) and re-adapt when the goal changed (*unexpected uncertainty*). This neurobiologically plausible model demonstrates how neuromodulatory systems can predict goals in uncertain domains and how attentional mechanisms can enhance the perception for that goal. See Chapter 3 for details.

1.2.3 Terrain Classification with a Reservoir-Based SNN

(This subsection is reprinted, with permission, from the abstract of a previously published work (Zou et al., 2020a). ©2020 IEEE.)

Terrain classification is important for outdoor path planning, mapping, and navigation. We developed a reservoir-based spiking neural network (r-SNN) to classify three terrain types (i.e. grass, dirt, and road) in a botanical garden. It included a recurrent layer and a supervised layer. The input spike trains to the recurrent layer were generated from linear accelerometer and gyroscope sensor signals as well as camera frames from an Android smartphone that controlled a ground robot. Compared to a Support Vector Machine (SVM) model and a 3-layer (3L) logistic regression model, our r-SNN method generated better prediction accuracy without reliance on a time window of data. Using both images and sensors as input, the test accuracy of the r-SNN was over 95%, which was significantly better than the SVM and the 3L logistic regression. Because the r-SNN is compatible with neuromorphic hardware, our proposed method could be part of a biologically-inspired power-efficient autonomous robot navigation system. See Chapter 4 for details.

1.3 Additional Projects

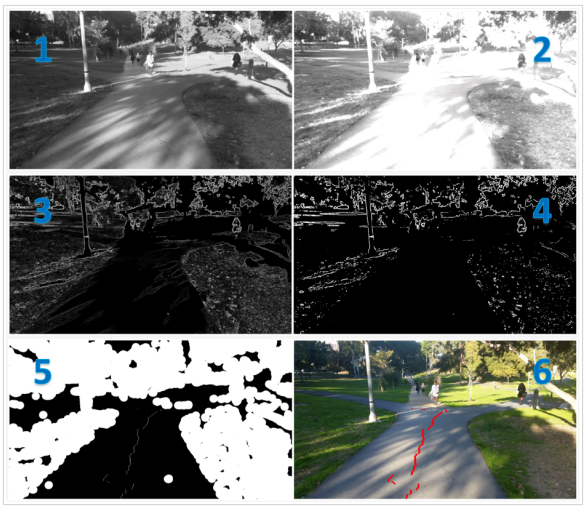
Aside from the three major projects, I also explored a few other research projects with my supervisor Jeffrey Krichmar and my labmates in the past five years. Most of them are related to autonomous navigational strategies using an Android Based Robot (ABR) (Oros and Krichmar, 2013). However, all these systems are applicable to more advanced robots and test setups.

The six-wheeled ABRs are generally used outdoor because of more intensive power and less friction. In the first side project, we designed a light-weight yet reliable road following

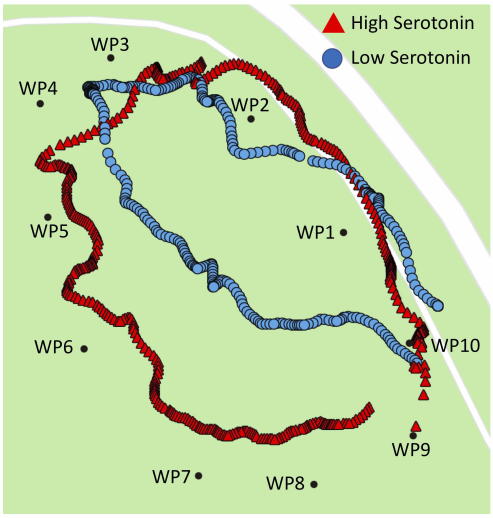
strategy using OpenCV to keep our six-wheeled ABR on the smooth road when planned regardless of distractions from shadows or obstacles (see Figure 1.7a). We also combined this reactive strategy with our spike-based path planning towards implementation of an energy-efficient, event-driven, massively parallel neuromorphic system for outdoor navigation (Hwu et al., 2017a). In the second side project, we added the rodent model of patience to waypoint navigation on an autonomous six-wheeled ABR (see Figure 1.7b). A higher seronin (5-HT) level led to more patient behavior of visiting more waypoints and taking fewer shortcuts (Xing et al., 2020).

The four-wheeled or tank ABRs are used indoor. In the third side project, we set up a prey-predator experiment with two four-wheeled ABRs. Our Q-learning-based foraging agent (see Figure 1.7c) significantly outperformed the random and actor-critic agents by obtaining more food, avoiding the predator, and not starving (Krichmar et al., 2019). In the fourth side project, we conducted schema consolidation for room recognition (see Figure 1.7d). Each time when a room flavor was detected from ground QR codes by the four-wheeled ABR, we would apply a self-organizing map (SOM) to update place-flavor paired associations. Our network connected a HPC layer of short-term pairing information to an mPFC layer which gradually learned schemas. Then it went through replay to detect if there was a schema/room switch or an flavor change in the existing schema/room. This project was later expanded to Hwu et al. (2020).

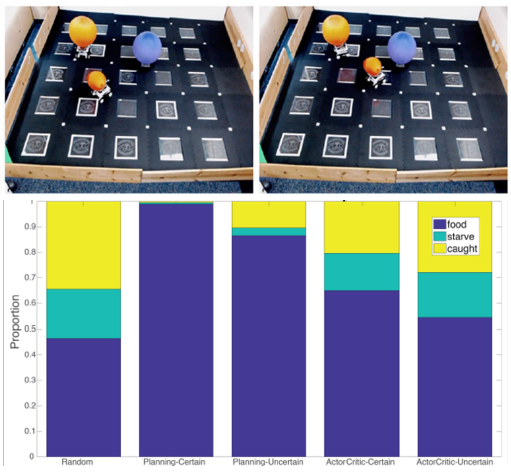
There was an additional simulation-based project which can be considered as an exercise before our cognitive map project (Zou et al., 2021). We implemented a DQN that could quickly adapt to the random relocation of a large reward in a double T-maze. We also tried to utilize the adaptive HyperNEAT to evolve connectivity patterns and plasticity rules encoded by CPPNs so that the network could learn the maze topology and plan the most efficient path towards a large reward.



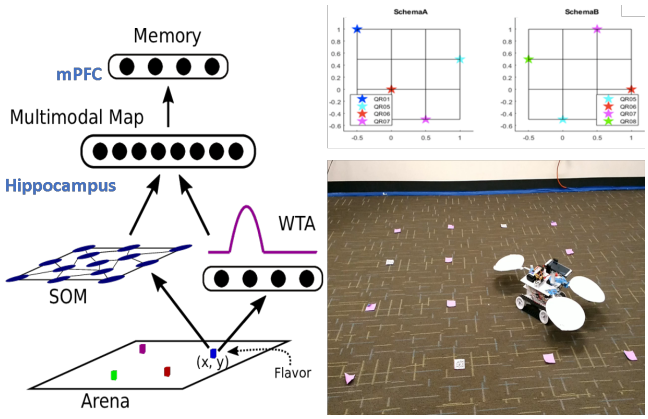
(a) Road Following



(b) Neuromodulated Patience



(c) Q-Learning-Based Foraging



(d) Schema Consolidation for Room Recognition

Figure 1.7: Additional projects in Xinyun Zou's doctoral research. (a–c) Images are adapted from (Hwu et al., 2017a; Xing et al., 2020; Krichmar et al., 2019). (d) The schema project was later expanded to (Hwu et al., 2020).

Chapter 2

Neuroevolution of a Recurrent Neural Network for Spatial and Working Memory in a Simulated Robotic Environment

(Except from the newly added section "Further Study on Latent Learning", this chapter is reprinted, with permission, from Zou, Xinyun, Eric O. Scott, Alexander B. Johnson, Kexin Chen, Douglas A. Nitz, Kenneth A. De Jong, and Jeffrey L. Krichmar. (2021). Neuroevolution of a Recurrent Neural Network for Spatial and Working Memory in a Simulated Robotic Environment. *arXiv preprint arXiv:1903.06070.*)

(A 2-page peer-reviewed poster version has been accepted in *Proceedings of 2021 Genetic and Evolutionary Computation Conference Companion (GECCO '21 Companion).*)

2.1 Introduction

The cognitive map, a concept raised by Edward C. Tolman in 1930s (Tolman, 1948), describes that the mental representation of a physical space could be built by integrating knowledge gained from the environmental features (e.g., goals, landmarks, and intentions). We used the Webots robot simulation environment (Michel, 2004) to investigate cognitive map behavior observed in rats during a spatial and working memory task, known as the triple T-maze (Olson et al., 2017, 2020). We suggest that similar behavior could be observed in a robot that had a biologically plausible neural network evolved to solve such a task. In this task, the rat or the robot must take one of four paths to receive a reward. If it repeated a path, there would be no additional reward. It would eventually learn to quickly reach each of the four rewards with minimal repeats. This requires knowledge of where it is now, where it has been, and where it should go next.

In our Webots setting, we designed a 3-D environment that resembled the rat experiment. The proximity sensors, the linear accelerometer and the grayscale camera pixels of a simulated e-puck robot (Mondada et al., 2009) provided sensory input for a recurrent neural network (RNN). The RNN output directly manipulated the motor speed of the e-puck. Using neuroevolution, the input weights into the RNN, the recurrent weights within the RNN, and the output weights from the RNN were evolved based on an objective designed to replicate the rat behavior.

Our results show that the evolved RNN was capable of guiding the robot through the triple-T maze with similar behavior to that observed in the rat. Our analysis of the RNN activity indicated that the behavior was not dependent on any one sensory projection type but rather relied on the evolved RNN dynamics. Furthermore, the population of neurons in the RNN were not only sufficient to predict the robot's current location but also carried a predictive code of future intended reward paths. Furthermore, the present method for evolving neural

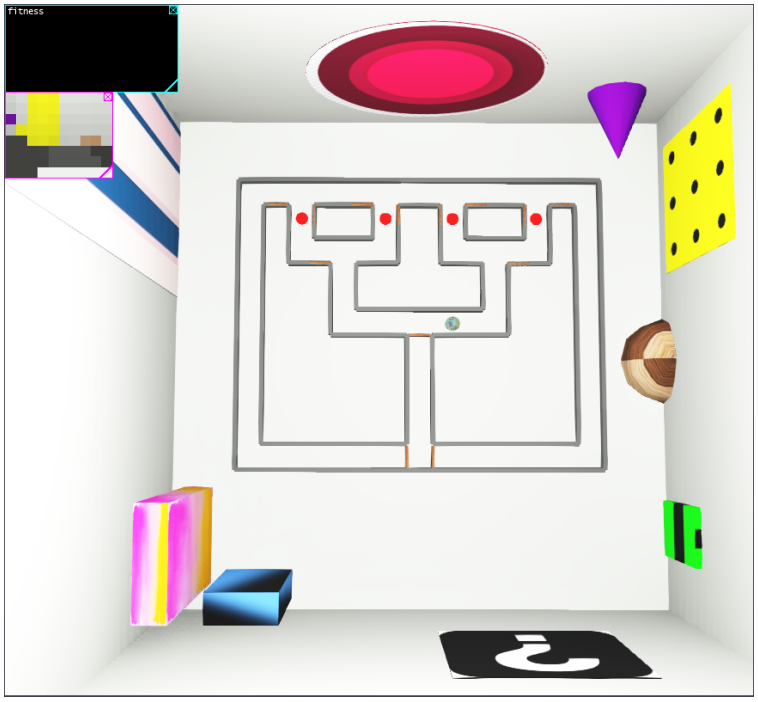


Figure 2.1: The maze visualization in Webots. As shown in the figure, there were corridors that the robot could traverse and landmarks on the wall. The e-puck robot is denoted by the small green circle. The red circles denote the reward locations. Note that these rewards were not visible to the robot's sensors.

networks for robot controllers may be applicable to other memory tasks.

2.2 Methods

We picked Webots (Michel, 2004) as our virtual robotic environment. Inside this 3D simulator, a triple T-maze was constructed that closely followed the dimensions and landmarks

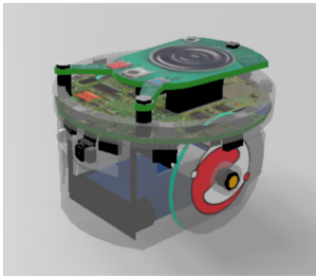


Figure 2.2: The 3D simulated e-puck robot. The picture is adapted from Webots (2020).

used in the rat experiment (Olson et al., 2017). Figure 2.1 shows the maze simulation environment. The red circles, which denote the location of the rewards, were not observable by the robot and are only included in the figure for illustrative purposes. The agent was an e-puck robot (Mondada et al., 2009) which has an accelerometer, a front camera, 8-direction proximity sensors, several LEDs, and 2 wheel motors (see Figure 2.2). The e-puck needed to learn by neuroevolution to find four rewards (and return home after each reward visit) with minimal repeats before timeout. Its actuation was updated every 64 milliseconds. The timeout threshold was tuned to 5000 steps (i.e., 320 seconds in real-time) per trial to guarantee enough time to visit all four rewards with minimal repeats and some tolerance for slight movement variations. For each trial, the robot would always start from the home position in the bottom middle part of the maze and move upward (see the e-puck's location in Figure 2.3). After reaching a T-intersection, a door behind the robot would close to prevent backtracking. Since the robot could only move forward in a reward path, it could neither revisit places on the same path nor switch to a different one before completion of the previous path. After the robot moved within 6 cm from the center position of a novel reward in a trial, the reward was added to the objective function given by Equation 2.3. After the robot passed through the third T-intersection right above any reward position, an additional door on the right/left would close to enforce its usage of the closer return path to home before exploring the next reward path. It should be noted that although the doors prevented backtracking, the robot still needed to evolve its ability to move smoothly and efficiently through the corridors to receive all four rewards with minimal repeats prior to the timeout.

The rotation speed of an e-puck was ranged between -3.14 rad/s and 6.28 rad/s. The evolved output weights in the RNN adjusted the speed and the turning rate of the robot to navigate the task with optimal and stable performance. To prevent from being stuck at corners or T-intersections, the robot had a default obstacle avoidance algorithm that used the 8 proximity sensors, which only influenced movement when the robot was very close to an

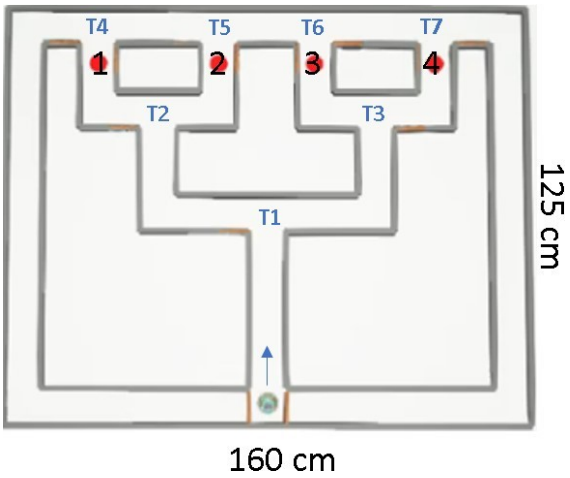


Figure 2.3: A closer look of the maze, with 4 rewards labeled in red circles and 7 T-intersections labeled in blue font. Home was located at the e-puck's current position (bottom-middle) in this figure.

obstacle, to move away from the closest point of contact (Fajen and Warren, 2003). The obstacle avoidance motor signal was added to the rotational speed of the motors dictated by the RNN output.

We conducted 5 evolutionary runs and selected the best performing agent from each run for further analysis. An evolutionary run was composed of 200 generations to achieve optimal performance. During each generation, there were 50 genotypes generated according to the evolutionary algorithm described in Section 2.2.1. For each genotype during the evolutionary process, the fitness value was recorded as an average over 5 trials to improve robustness in the selection. In each test scenario afterwards, each of the 5 best performing agents from these runs was utilized to run 20 demo trials with the same task setting and timeout threshold. For each demo trial, the activities of 50 recurrent neurons and the robot positions were recorded at each time step for further analysis.

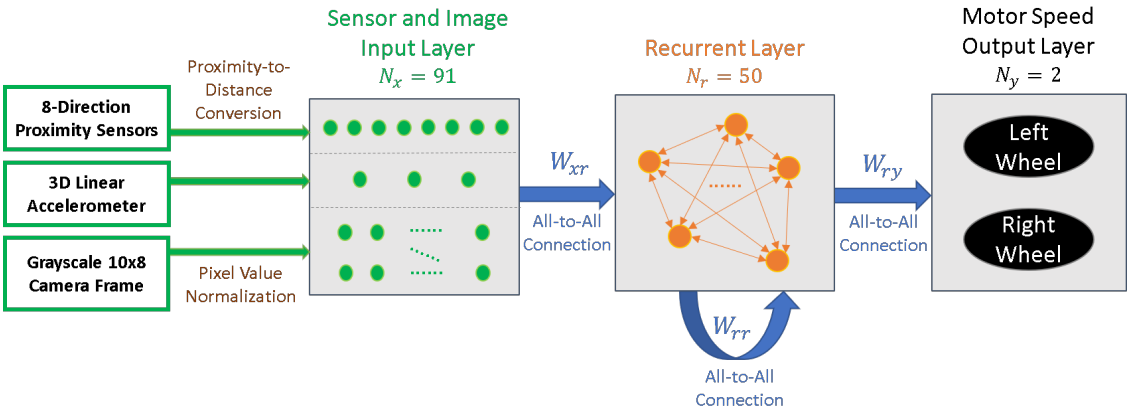


Figure 2.4: The neural network architecture for controlling the e-puck robot in Webots. Sensors were converted into input neural activities. The input weights (W_{xr}), recurrent weights (W_{rr}), and output weights (W_{ry}) were evolved concurrently. The output weights dictated the left and right rotational wheel speed of the e-puck.

2.2.1 Network Architecture

The neural network architecture received inputs from the e-puck's 8-direction proximity sensors, 3D linear accelerometer values, and normalized pixel values from its 10×8 grayscale camera frame (Figure 2.4). These 91 input neurons were fully connected to 50 recurrent neurons, which were fully connected with one another. This recurrent layer was then fully connected with the two neurons in the output layer that controlled the rotational speed of the two wheel motors separately.

Recurrent neural network

For each recurrent neuron i, its recurrent activity R_i was updated at every time step t according to Equation 2.1:

$$\left.\begin{aligned} R_i^{(t=0)} &= 0.0, \\ \text{synIn}_i^t &= \sum_{k \in in} W_{ki} \cdot x_k^t + \sum_{j \in rec}^{j \neq i} W_{ji} \cdot R_j^{(t-1)}, \\ R_i^t &= (1-p) \cdot \tanh\left(\text{synIn}_i^t\right) + p \cdot R_i^{(t-1)} \end{aligned}\right\} \tag{2.1}$$

Here x denoted the input sensor value and W_{ki} was the weight from Neuron k to Neuron i. In other words, the synaptic input for each recurrent neuron (synIn_i) contained (1) the summation of the product of each sensor value and the corresponding input weight plus (2) the summation of the product of every other recurrent neuron's previous activity and the corresponding recurrent weight. The tanh wrap ensured the recurrent activity between -1.0 and 1.0. A small p value of 0.01 helped to avoid any abnormal performance of the computed recurrent activity. The recurrent activity was then used to compute the rotational speed of each wheel motor by multiplying with the output weight.

Evolutionary algorithm

An evolutionary algorithm was used to evolve the input, recurrent, and output weights (W_{xr}, W_{rr}, and W_{ry} in Figure 2.4). Because of all-to-all connections, there were a total of 7150 genes for each genotype. The fitness value of each genotype was the average over 5 trials. The evolutionary algorithm used a population of 50 genotypes selected by linear ranking. Two-point crossover and mutation (with a decaying mutation standard deviation) were applied to reproduce the non-elite 90% of the population. The mutation rate was 0.06

and the mutation standard deviation decreased throughout a run via the function:

$$\text{mutation\_std} = 0.3 \times \frac{50.0}{50.0 + m} \tag{2.2}$$

Here m denotes the generation index.

The fitness function is shown in Equation 2.3:

$$\text{fitness} = \text{num\_obtainedRwds} + \text{portion\_routes\_completed} - 0.2 \times \text{num\_repeats} \tag{2.3}$$

During each trial, we would reward (1) each non-repetitive visit of any reward arm (i.e., $0 \sim 4$) and (2) the portion of reward path visits for which home was returned afterwards (i.e., $0 \sim 1$); meanwhile, we would penalize every repeated visit.

2.2.2 Bin-based Recurrent Activities

To analyze the performance of the robot after evolution, we divided the 1.6m-by-1.25m maze into 0.08m-by-0.10m sized bins, which was close to the e-puck's diameter (0.074m) (Figure 2.9). We computed the average activity of each recurrent neuron for each bin in the maze. For each of the 5 best performing agents (from 5 evolutionary runs), we used 15 demo trials to generate the expected bin-based activity matrix and 5 demo trials for bin occupancy prediction. The results are shown in Sections 2.3.4 and 2.3.5.

Spatial memory analysis

We used the RNN activity to predict the robot's location in the maze. Since there were 50 recurrent neurons, for each demo trial, we generated a bin-based recurrent activity matrix of size 110×50. The activity at each bin for each neuron was averaged over all steps spent on

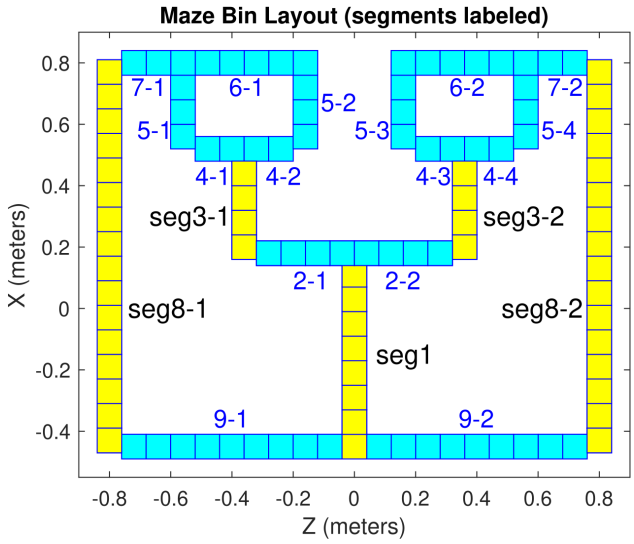


Figure 2.5: The maze layout with 110 bins of size 0.08m-by-0.10m. Segments used for the trajectory-dependent coding analysis are labeled as seg1, seg3-1, seg3-2, seg8-1 and seg8-2 in yellow.

that bin in a trial. Then we obtained an expected bin-based activity matrix of size 110×50 by taking the average over all the matrices for the first 15 demo trials. The remaining 5 trials were used to analyze the RNN's ability to encode location. For each of these 5 test trials, we compared each bin's RNN activity vector, which had a length of 50, with all 110 expected bin-based activity vectors using a Euclidean distance metric. The predicted bin was the smallest Euclidean distance to that actual bin. Thus, the Euclidean distance denotes the prediction error in bins. For example, if the Euclidean distance was 6, there was a perfect prediction error of 6 bins or approximately 0.5m (see Figure 2.10).

Trajectory-dependent analysis

After traversing some of the maze's vertical (South-to-North) segments, the robot would decide to turn left or right at a T-intersection. We analyzed if the RNN activity during traversal of a vertical segment could predict the robot's future path or the robot's prior path. As shown in Figure 2.5, Segment 1 is the vertical segment right before the first T-intersection on the path for any of the four rewards. Segment 3-1 (or Segment 3-2) denotes

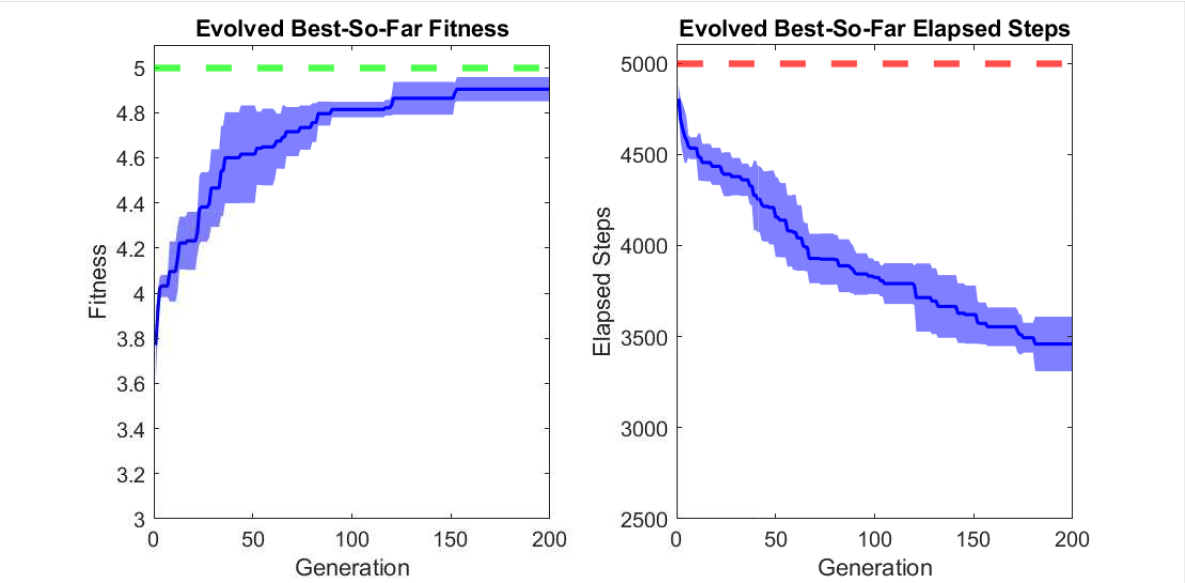


Figure 2.6: The evolutionary performance (left: fitness, right: number of elapsed time steps) for the best-so-far agent evolved in each generation. Each subplot was averaged over 5 runs with 200 generations per run. The shaded area denotes the 70% confidence level.

the vertical segment right before the second T-intersection on the path for Reward 1 or 2 (or for Reward 3 or 4). Segment 8-1 (or Segment 8-2) represents the vertical segment for returning from Reward 1 or 2 (or from Reward 3 or 4). Similar to the method described above, we computed an expected bin-based activity matrix for each of these segments from the first 15 demo trials for each reward path. We then used the remaining 5 demo trials to test whether the RNN activity could predict which path the robot was taking (i.e., *Prospective*; seg1, seg3-1, seg3-2) or which path the robot was returning from (i.e., *Retrospective*; seg8-1 and seg8-2).

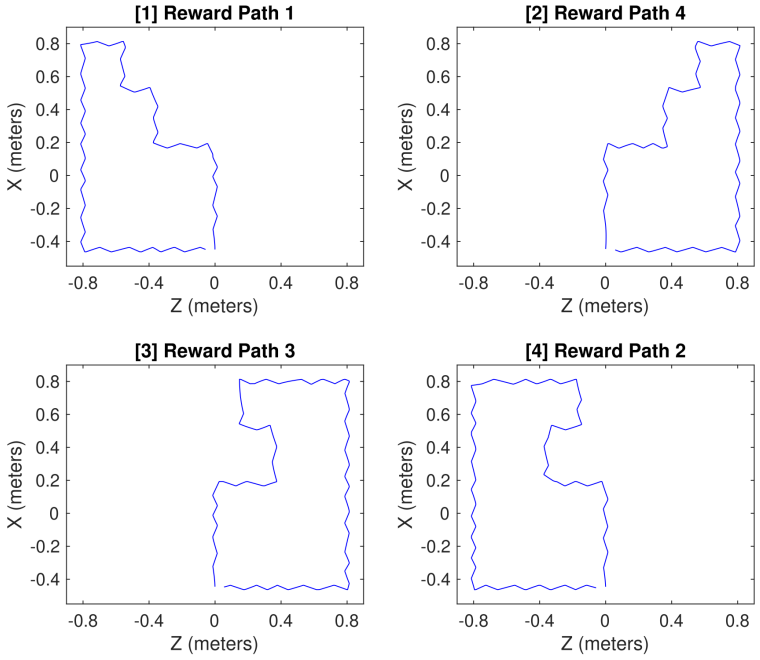


Figure 2.7: The trajectory of a perfect trial, which covered reward paths 1, 4, 3, 2 in order with no repeated visits of any path.

2.3 Results

2.3.1 Evolutionary Runs

Successful behavior, similar to that observed in rats, emerged from the evolutionary process. We ran all simulations using Webots (version R2020a) on a desktop with one GPU (Nvidia GeForce GTX 1080 Ti). Figure 2.6 shows the best-so-far evolutionary performance and the number of elapsed steps for five runs. Each run lasted 200 generations, with 50 genotypes per generation. In each generation, the fitness value of a genotype was averaged over 5 trials. By the end of each run, the best-so-far fitness curve reached a plateau close to the maximal fitness value of 5, whereas the number of steps taken to complete the task dropped below 3500 steps per trial on average. An example perfect trial trajectory (with no repeated visits of any reward path and a fitness of 5) can be observed in Figure 2.7.

Table 2.1: Ablation performance (mean ± the 95% confidence level) over 20 trials per ablation test for the best performing agent in each of the 5 evolutionary runs. The values highlighted with bold fonts and asterisks denote ablations that had a significant impact on the performance. Significance threshold was a p-value ¡ 0.01/6 = 0.0017 using the Wilcoxon Rank Sum test.

	Fitness	Elapsed Steps
No Ablation	3.65±0.14	4644±169
Proximity Sensors	3.29±0.16	4774±128
Linear accelerometer	3.49±0.16	4640±131
Grayscale Vision	3.21±0.17	4785±67
Input Weights	2.93±0.085	**4981±16***
Recurrent Weights	**2.95±0.11***	**4946±43***
Output Weights	**2.84±0.10***	**4978±22***

2.3.2 Ablation Performance

We carried out a set of ablation simulations to test whether performance was dependent on any sensory projection type or just evolved weights in the neural network (Figure 2.4). To test this, we either shuffled different sensor input values or shuffled the RNN input (W_{xr},) recurrent (W_{rr}), or output (W_{ry}) weights. For each of the 6 ablations, we ran the best performing agent from each of 5 evolutionary runs in demo trials. The results were averaged over 20 demo trials for each shuffle test. Random shuffle sequences occurred at each time step for each demo trial.

The ablation studies show that the dynamics of the RNN was critical for performance (Table 2.1). We compared the control (no ablation) with the 6 ablation groups. Since there were 6 comparisons, the significance threshold for the p-value is 0.01/6 = 0.0017 based on a Bonferroni correction. Interestingly, none of the sensory projection ablations had a significant impact on performance. However, ablating the evolved weights (input, recurrent, and output) all had a significant impact. That suggests that it was the recurrent neural network dynamics that were key to performance. Moreover, that performance was not dependent on any one sensory projection type.

2.3.3 Order Effects

We wanted to test if the robot evolved strategies to solve the triple-T maze task. Rats tend to show idiosyncratic behavior in the same maze setting (Olson et al., 2020, 2021). For instance, a rat alternated by going to the left side of the maze towards Rewards 1 and 2, and then the right side of the maze towards Rewards 3 and 4. Idiosyncratic behavior did emerge in our evolved robots. Although we did not observe the robots alternating between sides of the maze, each best performing agent for an evolutionary run exhibited a unique strategy for traversing the maze. Figure 2.8 shows the probability of transitioning from one reward path to the next. Only transitions probabilities that are greater than 0.33 are shown. We did find some generalities between genotypes. For example, the agents evolved in Genotypes 2, 3, and 5 tended to transition from the path for Reward 4 to the path for Reward 1. The agents evolved in Genotypes 2, 3, and 4 tended to transition from the Reward 3 path to the Reward 1 path. In both cases, the robot was navigating the right side of the maze before transitioning to the left side of the maze. These strategies that emerged in our robot and in the rat may simplify the task by breaking down the problem into chunks (e.g., first go to the right, and then go to the left).

2.3.4 Spatial Coding in the RNN

We investigated if the RNN activity was sufficient to predict the robot's position. If the activities of the 50 recurrent neurons could accurately encode the position, then the robot might be using this piece of information to solve the maze task. Borrowing techniques from neuroscience (Olson et al., 2017; Wilson and McNaughton, 1993a; Ferbinteanu and Shapiro, 2003), we tested whether the RNN contained spatial information with a population code.

Individual neurons in the recurrent layer did not seem to have place information. For example, Figure 2.9 shows the average bin-based recurrent activities across the entire maze

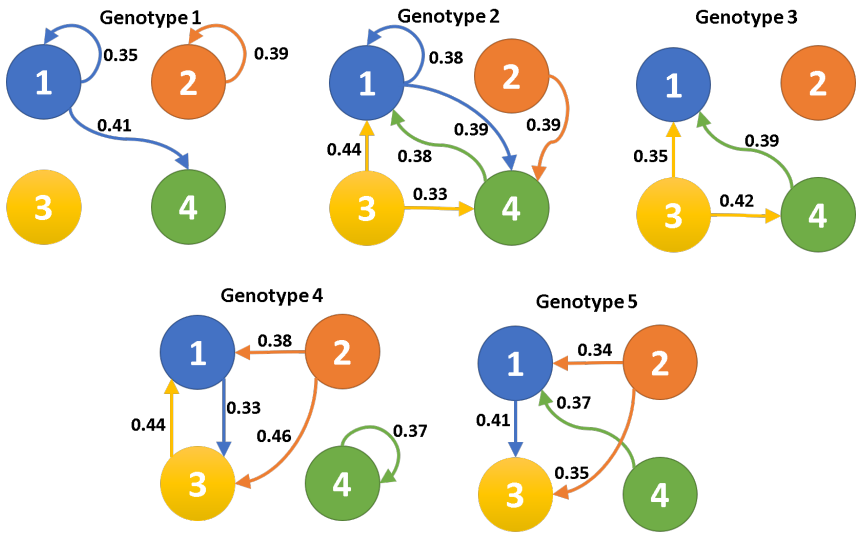


Figure 2.8: The trend of transitioning from one reward path to the next was different for each best performing agent (with a different genotype evolved) after 200 generations for each of the five evolutionary runs. The numbered circles denote the reward path, and the labeled arrows denote the probability of transitioning from one reward path to another.

from 15 demo trials of a top performing agent. Each bin's activity per trial divided by the number of time steps spent on that bin. The activity of each neuron is noisy with some neurons being highly active, quiescent, or oscillating.

However, the population of 50 RNN neurons was able to predict the robot's location throughout the maze. With the method described in Section 2.2.2, Figure 2.10 shows the location prediction for each bin in all 25 test trials (with 5 best performing agents from 5 evolutionary runs and 5 trials per agent). It is apparent from the figure that the RNN activity was sufficient to predict the robot's position in the maze. The robot's position was predicted with perfect accuracy on 58% of the bins, and the predicted error had an average distance of 3.1 bins (i.e., 0.25 meters).

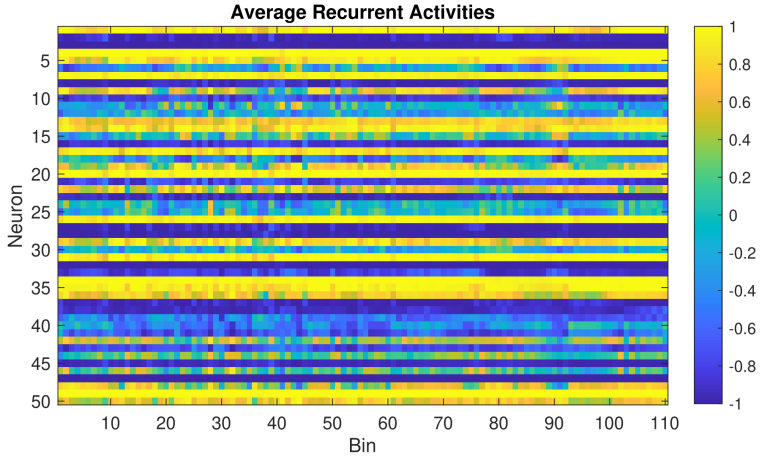


Figure 2.9: The average bin-based activities for all 50 recurrent neurons on 110 bins across the entire maze for a top performing agent.

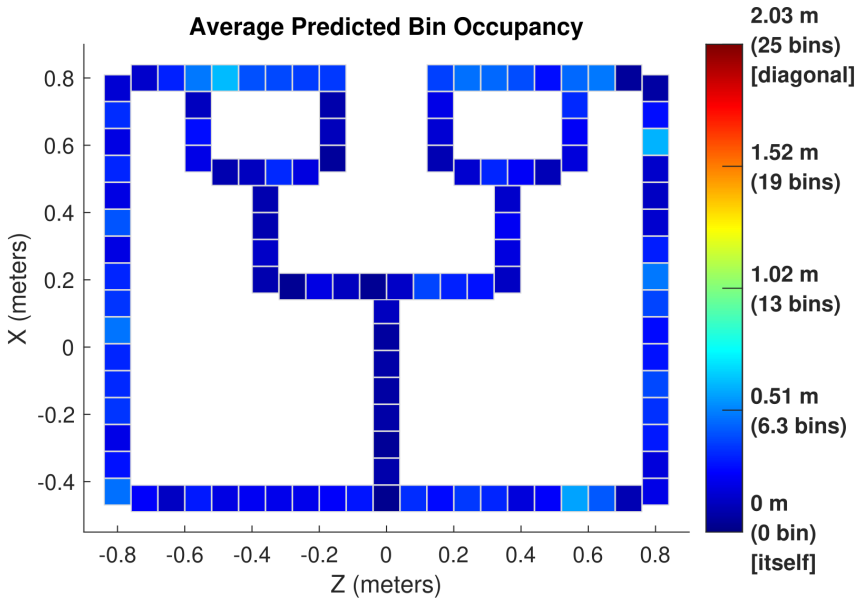


Figure 2.10: The average predicted bin occupancy over all 25 test trials of the best performing agent from each of the five evolutionary runs. A dark blue bin (if existing) would have perfect prediction right at itself (distance = 0), whereas a dark red bin (if existing) would have a farthest prediction (across the diagonal of the entire maze).

2.3.5 Trajectory-Dependent Coding in the RNN

Solving the triple-T maze task requires the agent, whether it is a robot or an animal, to remember which path it has already taken, as well as to decide which path to take next. We hypothesized that the dynamic activity of the RNN carried such information, which is known as retrospective coding (i.e., where it has been) and prospective coding (i.e., where it intends to go).

To test whether there was retrospective coding and/or prospective coding in the RNNs, we analyzed the RNN's ability to encode trajectory-dependent information at a population level. Table 2.2 shows how well the RNN could predict the robot's future path based on the activity of Segments 1, 3-1 and 3-2, and how well the RNN could predict the robot's past path based on the activity of Segments 8-1 and 8-2. The probability of correct path prediction on Segment 1, where the robot could take one of 4 paths, was well above chance level (t-test; p ¡ 0.0001). The correctness on Segment 3-1 or 3-2, where the robot could take one of two paths, was also well above chance level (t-test; p ¡ 0.0001 for Segment 3-1 and p ¡ 0.005 for Segment 3-2). This suggests that the RNN carried a prospective code of where the robot intent to go next. The probabilities of correct path prediction on Segments 8-1 and 8-2 were not above chance (t-test ¿ 0.05), which indicates they did not predict if the robot came from one of the 2 prior paths.

Taken together, these results suggest that the evolved RNN had prospective information in that it could be discerned which direction the robot would take before turning left and right. It is interesting that we were not able to observe retrospective information in the RNN population since some knowledge of where the robot had already visited was necessary for the observed performance.

Table 2.2: Average prospective path prediction for different segments.

	Seg1	Seg3-1	Seg3-2	Seg8-1	Seg8-2
correctness	41%	77%	69%	47%	55%
bins off	0.9	0.2	0.2	2	2

2.4 Discussion

The present work demonstrated that a robot controlled by an evolved RNN could solve a spatial and working memory task where the robot needed to navigate a maze and remember not to repeat paths it had taken previously. The RNN population activity carried spatial information sufficient to localize robot, and the RNN population activity carried predictive information of which path robot intended on taking. Behavior was dependent on RNN dynamics and not any particular sensory channel. The present method shows that complex robot behavior, using a detailed robot simulation, could be realized by evolving all weights of a RNN.

2.4.1 Evolved RNN with Spatial and Working Memory

We evolved a RNN to control a robot in a spatial and working memory task that replicated behavior and neural activities observed in rats (Olson et al., 2017). The robot was able to navigate the triple T-maze efficiently by reaching all four rewards with minimal repeats. Successful performance required the robot to have spatial knowledge and working memory of which rewards it had already visited. Prospective information, in which RNN activity predicted the robot's intention, emerged in the simulations. Although not observed in the analysis, the neural network must have had retrospective information to minimize repeating previously traversed paths.

At the population level, the evolved RNN activity has similar characteristics as the hip-

pocampus. It has been observed that the population activity of the CA1 region in the hippocampus can accurately predict the rat's location in a maze (Wilson and McNaughton, 1993a; Olson et al., 2021). Furthermore, journey-dependent CA1 neurons have been observed in the rat that can predict the upcoming navigational decision (Ferbinteanu and Shapiro, 2003). Similar to CA1, the RNN received speed, direction, and visual information as input, and combined these types of sensory information to construct a journey-dependent place code (Taube et al., 1990; Sargolini et al., 2006; Kropff et al., 2015; O'Keefe, 1976; Hafting et al., 2005; Sun et al., 2019; Potvin et al., 2007; Frost et al., 2020).

In Olson et al. (2021) and in other rodent studies, it has been observed that rats acquire individual strategies to solve navigational tasks. For example, in the triple-T task, many rats alternated between the left and right arms of the maze. Similarly, our robot demonstrated this idiosyncratic behavior. Each best performing agent for an evolutionary run had an order-dependent pattern for taking different paths in the maze. It suggests that the RNN evolved a strategy breaking down the complex maze task into simpler pieces, which may also be how animals solve tough problems.

2.4.2 Behavioral Dependence on RNN Dynamics

To investigate the dependence of the robot behavior on different components of the RNN neural architecture, we conducted an ablation study (see Figure 2.4). Results in Section 2.3.2 show that ablating a given sensory channel had no significant impact on performance. However, ablating the evolved weights (input, recurrent, and output) all had a significant impact. This suggests that it was the recurrent neural network dynamics that were key to performance, and that performance was not dependent on any one sensory projection type. The results justify evolving all weights in the RNN structure, rather than evolving only the readout weights as is often done in Liquid State Machines (LSM) or Echo State Networks

(Maass et al., 2002). Furthermore, the present results demonstrate the potential of extending our RNN controller to other types of robots that use different sensory inputs.

2.4.3 Capability of Evolving Complex Robotics Behavior

One advantage of our method is the simple design of our fitness function. It only includes rewards for each visit of a novel reward and the completeness of each reward path plus a small penalty for repeated visits. We also tried with an additional reward term for the portion of time steps left before timeout, but finally excluded it from the fitness function. Instead, the time cost would be automatically influenced by rewarding non-repetitive reward path visits as demonstrated in Figure 2.6. It is not a fitness function only for a certain type of robot, because it is independent of robot properties (e.g., sensors, speed, motor structure, etc.). Therefore, it could be easily generalize to fit other task settings or robots.

The key to performance in our experiments was the evolved weights into, within, and from the RNN. Generally a RNN has connections between internal neurons form a directed cycle. Arbitrary sequences of inputs could be processed by using the internal state of the RNN as the memory. RNNs have been applied to a broad range of domains such as terrain classification, motion prediction, and speech recognition (Zou et al., 2020a; Kashyap et al., 2018; Graves et al., 2013). A reservoir-based approach, such as an LSM (Maass et al., 2002), can tractably harness such recurrence.

Similar to an LSM, we also tried a reservoir-based approach to evolve only the output weights while keeping randomly initialized input and recurrent weights fixed throughout generations. In addition, we attempted to evolve both output weights along with input or recurrent weights (but not with both). However, their evolutionary performance could not generate fitness values as well as evolving all three types of weights. Evolving all three types of weights allow maximal utilization of neural activities to create the dynamics needed to

solve a sequential memory task, such as the triple-T maze one. This process of evolving input, recurrent, and output weights could be readily transferred to other complex robotic settings.

Because of the accurate representation of many popular robot designs in the Webots simulator, we could always run much faster than real-time (e.g., $\times 30$ faster on average on our desktop with one GPU) to evolve for enough generations before applying a well-performed genotype to the RNN controller for real-world robotic navigation tasks. The power of using a detailed simulator, such as Webots, is that the evolved controller should transfer to the real e-puck with minimal adjustments.

2.4.4 Comparison with Prior Work

Evolutionary robotics

Evolutionary robotics is a method for building control system components or the morphology of a robot (Bongard, 2013; Nolfi et al., 2016). The biological inspiration behind this field is Darwin's theory of evolution, which was constructed with three principles. (1) *Natural Selection*: genotypes that can be well adapted to their environments are more likely to survive and reproduce. (2) *Heredity*: the fittest genotypes from the previous generation can be directly kept in the next generation; moreover, the new offspring in each generation is generated based on the selected genes from two parents. (3) *Variation*: the new offspring goes through mutations and crossover with a certain probability and thus differs from both parents.

Our method follows these three principles and falls into the common category of evolving the control system. However, what we evolved is novel compared to other work on evolutionary robotics. There has been work to evolve robots in cognitive tasks. For example, one group

evolved virtual iCub humanoid robots to investigate the spontaneous emergence of emotions. Their populations were evolved to decide whether to "keep" or "discard" different visual stimuli (Pacella et al., 2017). There has also been work on evolving robot controllers capable of navigating mazes. For example, Floreano's group evolved neural network controllers to navigate a maze without colliding into wall. Their neural networks were directly tested on a Khepera robot that had proximity sensors and two wheel motor system that was similar to the e-puck used in our present studies. Their evolved neural networks developed a direct mapping from the proximity sensors to the motors (Floreano and Keller, 2010). Our present work extends this prior work by evolving an RNN capable of navigating mazes, as well as demonstrating cognitive behavior.

Rather than evolving a direct mapping from sensors to motors, we instead evolved the weights from sensory inputs to the RNN, within the RNN, and from the RNN to the robot's motors as the genes for each genotype evolved in our network. As is discussed below, RNNs such as the ones we discuss here are neurobologically plausible and allow for comparisons with neuroscience and cognitive science data (Wang et al., 2018; Yang et al., 2019). Moreover, the RNN architecture is more generalizable to different types of robots working in complex scenarios (e.g., the triple T-maze with multiple rewards and landmarks), which results in optimal performance independent of any projection type.

Evolved RNNs

Although there are only a few studies that have RNNs evolved directly in robotic experiments, evolving RNNs has been more frequently applied to virtual task settings. For example, Akinci and Philippides (2019) used either a steady-state genetic algorithm (SSGA) or an evolutionary strategy (ES) to evolve weights of the Long Short-Term-Memory (LSTM) network or RNN for the Lunar Lander game provided by the OpenAI gym (Brockman et al., 2016). In their case, the ES developed more dynamic behavior than the SSGA, whereas the

SSGA kept good genotypes and re-evaluated them with different configurations.

Li and Miikkulainen (2018) evolved poker agents called ASHE with various types of evolutionary focus, such as learning diversified strategies from strong opponents, learning weakness from less competitive ones, learning opposite strategies at the same time, or a mix in-between. The genes in their GA covered all parameters in the estimators, including the LSTM weights, per-block initial states, and the estimator weights.

A more biologically inspired example is related to the recent work by Wieser and Cheng (2020). Inspired by the neuroplasticity and functional hierarchies in the human neocortex, they proposed to use a network called EO-MTRNN to optimize neural timescales and restructure itself when training data underwent significant changes over time.

All these related works have their unique perspectives that could inspire us to build a more robust and potentially faster evolutionary process for RNN systems in the future. For instance, we may consider to experiment with features in different evolutionary algorithms or co-evolve different neural regions which have different strategies or focus on a cognitive task.

Comparison with NEAT/HyperNEAT robot controllers

The evolutionary algorithms were utilized to evolve only weights in our RNN system and many other groups' work as mentioned earlier. Another popular evolutionary mechanism is NEAT, which has its unique feature of evolving the network topology together with the weights (Stanley and Miikkulainen, 2002). HyperNEAT extends NEAT by evolving connective CPPNs that generate patterns with regularities (e.g., symmetry, repetition, repetition with variation, etc.) (Stanley et al., 2009). In the case of quadruped locomotion investigated by Clune et al. (2009), HyperNEAT could evolve common gaits by exploiting the geometry to generate front-back, left-right, or diagonal symmetries. Our current model was tuned to have a fixed number of recurrent neurons since the first generation and have all-to-all con-

nections between two layers or within the recurrent layer. It may be of interest to combine NEAT/HyperNEAT with topologies of RNNs to solve more complex problems and scenarios. For example, NEAT might be utilized in the beginning of the evolutionary process to efficiently derive a morphology for a more standard evolutionary algorithm to use in later generations. This hybrid approach has similarities to Akinci and Philippides (2019).

Working memory

Our evolved RNN encoded not only spatial information but also working memory to remember which paths had been traversed recently and which paths remained to be explored. Working memory helps to connect what happened earlier with what occurs later. It can be thought of as a general purpose memory system that can generalize, integrate and reason over information related to decision making or executive control (Diamond, 2013; Vyas et al., 2020). For example, Yang et al. (2019) trained single RNNs to perform 20 tasks simultaneously. Clustering of recurrent units emerged in their compositional task representation. Similar to biological neural circuits, their system could adapt to one task based on combined instructions for other tasks. Furthermore, individual units in their network exhibited different selectivity in various tasks.

Working memory usually relies on the prefrontal cortex (PFC) for information maintenance and manipulation (Baddeley and Hitch, 1994; Eldreth et al., 2006; Smith and Jonides, 1999). Wang et al. (2018) investigated such brain functioning with a meta-reinforcement learning (meta-RL) system. Their model trained the weights of an RNN centered on PFC through a reward prediction error signal driven by dopamine (DA). This RNN "learned to learn", which means it had the ability to learn new tasks via its trained activation dynamics with no further tuning of its connection weights.

With further investigation and utilization of working memory, we also would like to have our

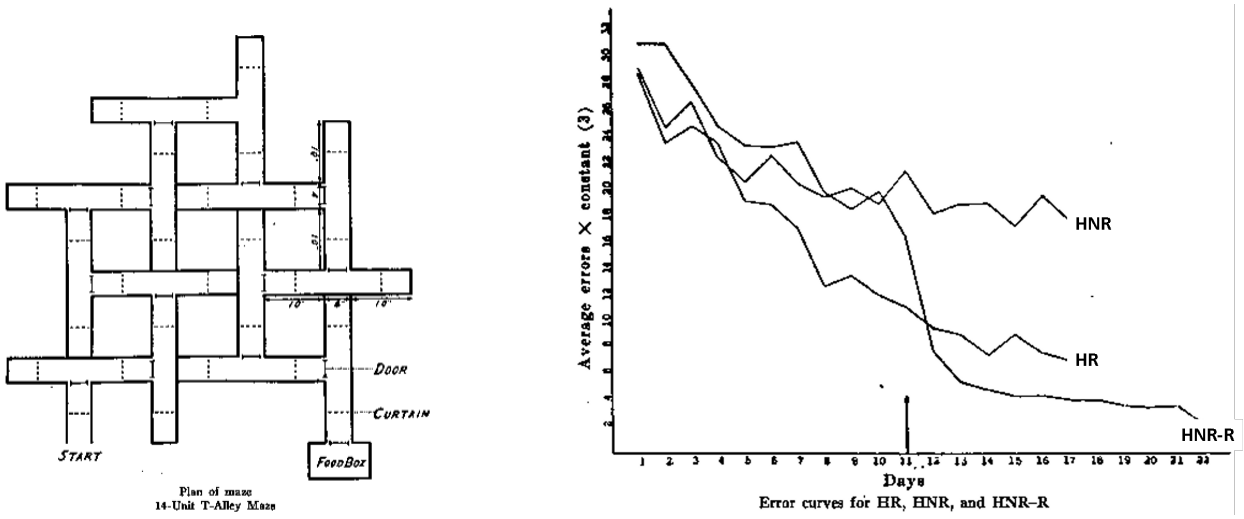
evolved RNN generalize over multiple cognitive tasks and demonstrate cognitive functions observed in different brain regions.

2.5 Further Study on Latent Learning

(This section is not yet included in any preprint or published material.)

One of Tolman's cognitive map ideas is "latent learning", which states that learning can occur in the absence of rewards, but does not manifest itself until a reward was introduced into the environment. In their original experiment (see Figure 2.11), one group of rats, which were fed with food at the maze end since the first day, did not learn to reach the goal-box as fast as the latent learning group did when food was introduced after several days of exploring (Tolman, 1948). It shows that rats could actively learn by building a mental representation of the space during exploration, even if no reward was presented. In other words, an animal can keep learning but do not show the learning result in their behavior until there is a motivation to demonstrate. Once the reward appears in the explored environment, its knowledge about the information accumulated in the cognitive map will be immediately expressed.

We investigated this idea in our simulated triple T-maze. We still evolved 50 genotypes per generation, but reduced the total number of generations per run from 200 to 100. Each run was composed of half generations without food rewards followed by half generations with food rewards. In the first 50 generations, each trial was initialized with no food rewards, and maze exploration was encouraged. The agent was still required to follow the rule of home→top arm→home when visiting each unrewarded route with minimal repeats. Therefore, only the term "num_obtainedRwds" in the original fitness function (see Equation 2.3) was removed for this period, and the timeout threshold for each trial was 5000 steps. In the second 50

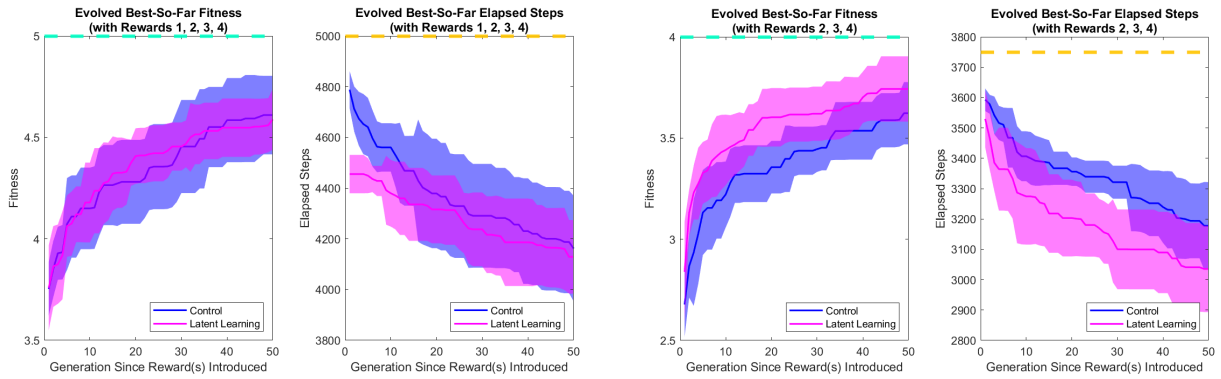


(a) Tolman's original maze (b) Tolman's original results

Figure 2.11: Tolman and his colleagues' original latent learning experimental setup and results. (a) The maze included 14 turning points (with doors) and 14 dead ends. (b) They used two control groups – one that never found food in the maze (HNR) and one that found it throughout (HR). The experimental group (HNR-R) found food at the end of the maze from the 11th trial on and showed the same sort of a sudden drop. The figure is adapted from Tolman and Honzik (1930).

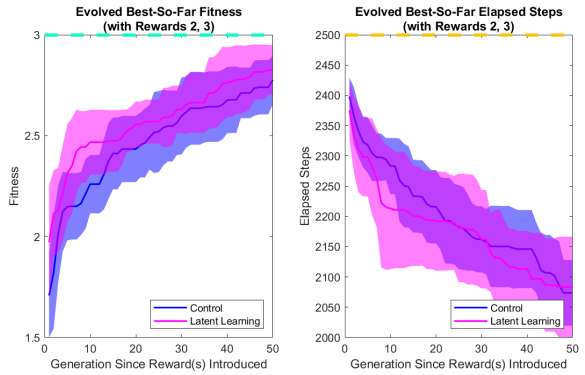
generations, each trial was initialized with one to four food rewards. The agent could still visit unrewarded top arms; however, only non-repeated visits increased fitness. The original fitness function in Equation 2.3 could be re-applied here, and the timeout threshold for each trial was reduced to be the product between the number of introduced food rewards and 5000/4 steps.

To evaluate the benefits of latent learning on the evolved performance, we compared each of the four latent learning cases with a corresponding control case. The control case had one to four food rewards introduced to each trial throughout all the generations. Figure 2.12 illustrates the best-so-far curves of control vs. latent learning experiments for all four reward cases. Results were compared between the first 50 generations of the control experiment and the second 50 generations of the latent learning experiment since the beginning of each run and averaged over 10 runs. For the best-so-far fitness, better performance would be associated with a steeper slope magnitude, a higher Y-axis value, and a larger area-under-

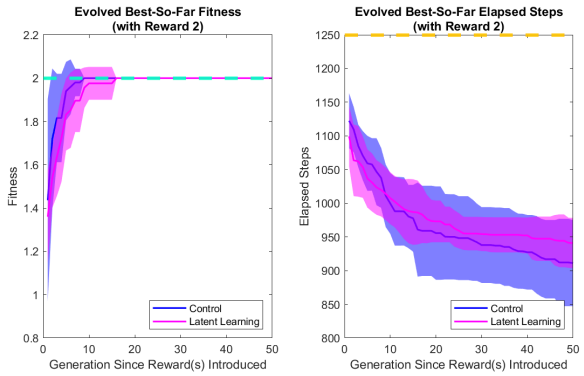


(a) with all four rewards (10 runs/case)

(b) with three rewards (10 runs/case)



(c) with two rewards (10 runs/case)

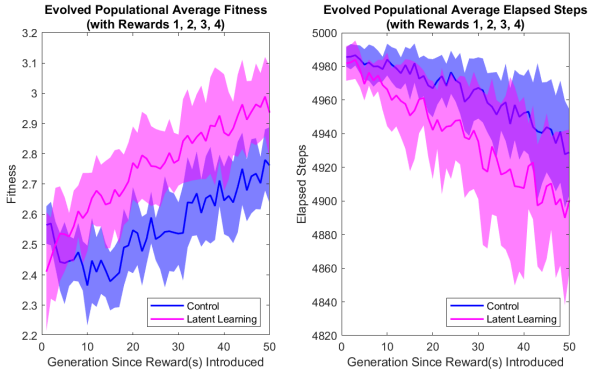


(d) with one reward (10 runs/case)

Figure 2.12: Best-so-far evolved performance of control v.s. latent learning experiments. The x-axis labels the generation indices *since the food reward(s) was/were introduced in each run*. For each run, the best-so-far performance at Generation i was linked with the best evolved genotype among all the past generations. A better performance is generally associated with a higher fitness or a lower number of elapsed steps. Each case is plotted as an average over 10 runs (with 1 standard deviation labeled in shaded regions). The dashed lines represent the maximum thresholds.

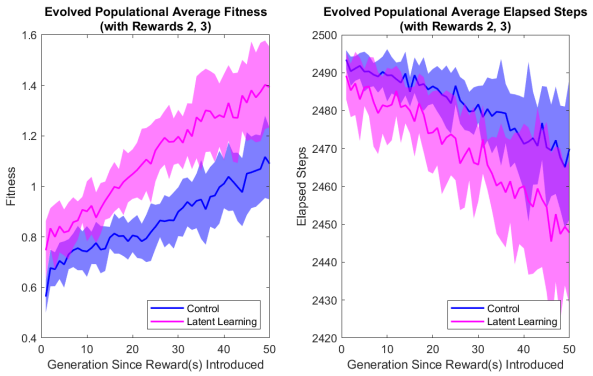
curve (AUC). For the best-so-far elapsed steps, better performance would be associated with a steeper slope magnitude, a lower Y-axis value, and a smaller AUC.

For the cases with two or three food rewards, the best-so-far curves for the latent learning experiment approached the optimal much faster than the control and contained better Y values. Such advantage for latent learning was not obvious to the case with all four rewards, because the encouragement of visiting all four food rewards was similar to that of exploring the entire maze. It was also not obvious to the case with only one reward, because the agent



(a) with all four rewards (10 runs/case)

(b) with three rewards (10 runs/case)



(c) with two rewards (10 runs/case)

(d) with one reward (10 runs/case)

Figure 2.13: Populational average evolved performance of control v.s. latent learning experiments. The x-axis labels the generation indices *since the food reward(s) was/were introduced in each run*. For each run, the populational average performance at Generation i was averaged over all 50 genotypes in the same generation. A better performance is generally associated with a higher fitness or a lower number of elapsed steps. Each case is plotted as an average over 10 runs (with 1 standard deviation labeled in shaded regions).

already achieved nearly perfect performance at 10 generations after the food reward was introduced in both control and latent learning experiments.

Aside from the best-so-far curves, we also plotted the populational average curves over 50 generations since the food reward(s) was/were introduced (see Figure 2.13). For all four reward cases, the latent learning demonstrated better populational average performance as indicated by its AUCs for both fitness and elapsed steps.

The initial results from the latent learning study are promising. We are close to the conclusion

that the agent which does not display its learning effect during maze exploration until there is some motivation can eventually perform better than the other agent which directly receives some stimulus since the beginning. We also need to analyze the recurrent neural activities for the latent learning experiment, especially at three stages: (1) at the end of the latent learning phase (i.e., Generation 50) to understand how working and spatial memory was better encoded when the agent was encouraged to explore the entire maze with no food rewards, (2) a few generations after introduction of food rewards (i.e., Generation 60) and (3) at the end of each run (i.e., Generation 100) to understand how the activities changed with respect to the new food stimuli. By doing so, we may be able to generalize the benefits of evolved latent learning to other spatial navigation scenarios. It might also be helpful to conduct benchmark comparisons between our neuroevolutionary method and state-of-art reinforcement learning models (e.g., DQN, A3C).

2.6 Conclusions

In this chapter, we demonstrated that an evolved RNN that controlled a robot could demonstrate aspects of cognitive map behavior. We introduced a recurrent neural network (RNN) model that linked the robot sensor values to its motor speed output. By evolving weights from sensory inputs to the RNN, within the RNN, and from the RNN to the robot's motors, the evolved network architecture achieved the goal of successfully performing a cognitive task that required spatial and working memory. The RNN population carried spatial information sufficient to localize robot in the triple T-maze. It also carried predictive information of which path robot intended on taking. Moreover, the robotic behavior was dependent on RNN dynamics rather than a sensor-to-motor mapping. Our method shows that complex robot behavior, similar to which being observed in animal models, can be evolved and realized in RNNs.

Chapter 3

Neuromodulated attention and goal-driven perception in uncertain domains

(This chapter is reprinted, with permission, from Zou, Xinyun, Soheil Kolouri, Praveen K. Pilly, and Jeffrey L. Krichmar. (2020b). Neuromodulated attention and goal-driven perception in uncertain domains. *Neural Networks, 125*, 56-69. *©2020 Elsevier Ltd.*)

3.1 Introduction

Artificial attentional mechanisms in neural networks tend to respond to sensory inputs similarly regardless of context and goals (Zhang et al., 2018; Itti and Koch, 2000; Tsotsos et al., 2015). However, biological systems select relevant information to guide behavior in the face of noisy and unreliable signals, as well as rapidly adapt to unforeseen situations. Goal-driven perception treats the same situation differently based on context and effectively directs at-

tention to goal-relevant inputs. Often, these goals are unknown and must be learned through experience. Moreover, these goals or contexts can shift without warning. Goal-driven perception helps prevent overemphasis on less relevant stimuli and instead focus on critical stimuli that require an immediate response.

In the brain, neuromodulators are important contributors to attention and goal-driven perception. In particular, the cholinergic (ACh) system drives bottom-up, stimulus-driven attention, as well as top-down, goal-driven attention (Avery et al., 2014). Furthermore, the ACh system increases attention to task-relevant stimuli, while decreasing attention to distractions (Baxter and Chiba, 1999; Oros et al., 2014). This procedure is similar to the core idea behind contrastive Excitation Backprop (c-EB). In c-EB, a top-down excitation mask increments attention to the target features, and an inhibitory mask decrements attention to distractors (Zhang et al., 2018). The noradrenergic (NE) system responds to surprises or large deviations from priors (Yu and Dayan, 2005). When the NE system responds phasically, where the neural activity rapidly and transiently increases, it causes a network to reset (e.g., re-initializing activities) that allows rapid adaptation under unseen/new conditions (Bouret and Sara, 2005; Grella et al., 2019).

We modified a c-EB network for use in a goal-driven perception task, where the system had to increase attention to the intended goal object and decrease attention to the distractor. In the first experiment, we presented pairs of noisy MNIST digits to the neural network. One goal class was to attend to the digit based on its parity (i.e., even or odd goal), and another goal class was to attend based on the magnitude of the digit (i.e., low- or high-value goal). In addition, we added a neuromodulatory model to the head of the network architecture that regulated goal selection. Similar to the model of the ACh and NE neuromodulatory systems proposed by Yu and Dayan (2005), we framed the task as an attentional task where the goal (even, odd, low or high value) had to be learned from experience (*goal identity*) and the goal might be noisy and rewarded with some probability (*goal validity*). In the second

experiment, we generalized our model to an action-based attention scenario, where "eat", "work-on-computer", "read", and "say-hi" were goal actions and the robot needed to attend to and retrieve objects that corresponded to the action.

3.2 Methods

We modified the c-EB neural network model to attend to different goals. Section 3.2.1 describes how we tested the ability of the network to increase attention to different goals and digits for the noisy MNIST-pair experiment. Section 3.2.2 introduces our neuromodulatory learning system to predict unknown and uncertain goals based on experiences, still using the noisy MNIST-pair experiment as an example. Section 3.2.3 describes how our method was generalized to demonstrate goal-driven perception in a human support robot.

3.2.1 Network Architecture

Figure 3.1 shows our bottom-up classification process and our top-down attentional search process. In the forward pass, the input layer received a pair of 28×28-pixel noisy MNIST digits and thus had $28 \times 28 \times 2 = 1568$ neurons (LeCun et al., 1998). To test the network's ability to filter out distractions, noise that was randomly set between 0 and 0.7 was added to normalized pixel values (between 0 and 1) of the original MNIST digits. The final pixel values were then normalized again between 0 and 1.

Following the input layer were two sequential fully connected hidden layers with 800 and 600 neurons, respectively. Next, there were two parallel fully connected hidden layers, each with 400 neurons. All neurons in these layers implemented a Rectified Linear Unit (ReLU) as the activation function (Nair and Hinton, 2010). Each of the two parallel hidden layers led to the output in one goal class (parity/magnitude) along with the digit output. For each

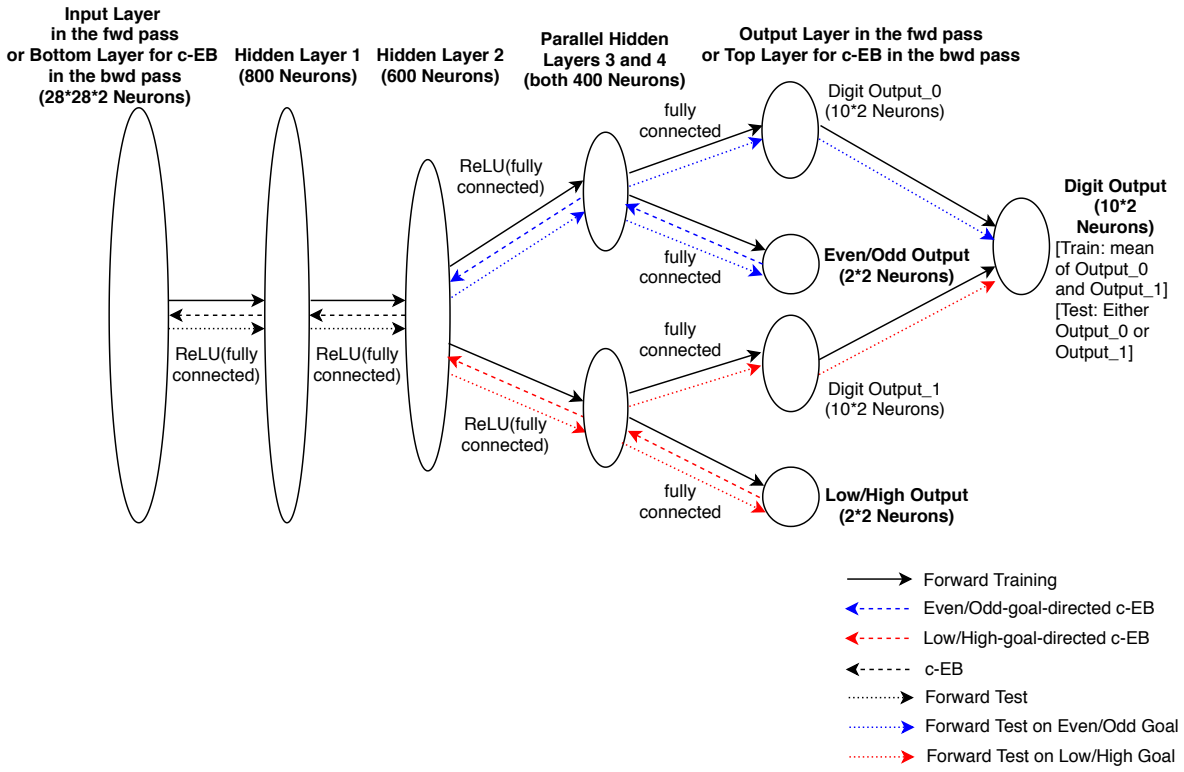


Figure 3.1: Network setup for our bottom-up classification process and our top-down attentional search process, with a pair of noisy MNIST digits as the input data in the forward pass.

(left/right) side of the input image, after the third hidden layer, there were two parity (an even and an odd) output neurons and ten digit output neurons, which contributed to the parity/digit prediction using winner-take-all (WTA) on the activation probability of each parity/digit output neuron. Similarly the magnitude/digit prediction was obtained after the fourth hidden layer for each side of the input image. During training, the final digit output took the average of the digit output generated from the two parallel hidden layers. During testing, the final digit prediction was the digit output generated from one of the two parallel hidden layers, depending on the cued goal class.

In our top-down attentional search process (see backward arrows in Figure 3.1), one of the four goals (even, odd, low, and high) was selected, which excited the corresponding goal neuron for each of the two digits in the image and inhibited all the other goal neurons at the top layer of the backward pass (i.e., the output layer of the forward pass). The weights were backpropagated from the top layer to one of the parallel hidden layers below to excite the neurons corresponding to the goal (see dashed arrows from the top layer to parallel hidden layers 3 and 4 in Figure 3.1). Then the weights at the top layer were converted from excitatory to inhibitory in order to create a mask (note that the weights were originally non-negative). This inhibitory mask was used in an additional backpropagation from the top layer to the parallel hidden layer below corresponding to the goal. The result of a subtraction between the two backpropagations was a contrastive signal (Zhang et al., 2018). This contrastive signal was then used to perform regular EB over the remaining layers, which finally generated the probability of each given pixel in the input layer for exciting the cued goal neurons. In addition to exciting the goal neurons and inhibiting non-goal neurons, the contrastive signal canceled out common winner neurons. Such a contrastive extension of the backpropagation could effectively ignore noisy distractors and lead to more accurate attention focus on the goal (Zhang et al., 2018).

Modification of c-EB

Excitation Backprop (EB) was developed as a goal-driven attentional framework for a CNN classifier based on a probabilistic winner-take-all (WTA) process (Zhang et al., 2018). It could visualize the features at each layer in the hierarchy that were relevant to a given output neuron. An important extension of EB was to have contrastive Excitation Backprop (c-EB), which discriminated the goal pixels from distractors by cancelling out common winner neurons for different goals and amplifying discriminative neurons for the target goal (Zhang et al., 2018).

The EB mechanism kept non-negative weights between activation neurons and used these excitatory connections to transmit top-down signals. The top-down relevance of a neuron a_n in the layer L_l was defined by its probability of being chosen as a layer-wise winner, which was called the Marginal Winning Probability (MWP) $P(a_n)$ (Zhang et al., 2018):

$$P(a_n) = \sum_{a_m \in (L_0, L_1, \ldots, L_{l-1})} \left(P(a_n|a_m) \cdot P(a_m) \right), \tag{3.1}$$

where a_m denoted each parent neuron in the preceding layer(s). The winner neurons were recursively sampled in the top-down direction according to the conditional winning probability $P(a_n|a_m)$ (Zhang et al., 2018):

$$P(a_n|a_m) = \begin{cases} \dfrac{\widehat{a_n} \cdot w_{nm}}{\sum_{n:w_{nm} \geq 0} (\widehat{a_n} \cdot w_{nm})} & \text{if } w_{nm} \geq 0, \\ 0 & \text{otherwise,} \end{cases} \tag{3.2}$$

where w_{nm} was the weight between a parent neuron a_m and one child neuron a_n, and $\widehat{a_n}$ denoted a non-negative activation response.

For c-EB, the contrastive signals were transmitted in the top-down fashion to obtain highly distriminative attention maps (i.e., contrastive MWP maps) in the target layer (i.e., the

bottom layer in Figure 3.1). Extending from Equation 3.1, Zhang et al. (2018) defined the contrastive MWP (c-MWP) of the target layer L_l as

$$A - \overline{A} = P_0 \cdot (P_1 - \overline{P_1}) \cdot P_1 \cdot ... \cdot P_{l-1}, \tag{3.3}$$

where A represented the MWP, $\overline{A}$ was the dual MWP for the contrastive units, P_0 was the signal from the guessed goal, $\overline{P_1}$ was the conditional probability of the inhibition mask from the top layer. The weights for the inhibition mask were the negation of the original weights from the top layer. Therefore, the threshold condition for $\overline{P_1}$ was the reverse of that for P_1 in Equation 3.2.

We extended the PyTorch (Paszke et al., 2017) implementation of c-EB (Greydanus, 2018), whereas the original code for c-EB (Zhang et al., 2018) was written in Caffe (Jia et al., 2014). Different from their implementation, our network included different goal classes labeled for each of the two noisy MNIST digits. In addition, the c-EB was processed through one of the two parallel hidden layers immediately below the top layer in the backward pass of our network.

The system increased attention to the digit corresponding to the selected goal and decreased attention to the distractor digit. One goal class attended to the digit based on its parity (i.e., either odd or even), whereas the other goal class attended to the digit based on its magnitude (i.e., low values between 0 and 4 inclusively or high values between 5 and 9 inclusively). This resulted in two goals within each goal class. After supervised training on noisy pairs generated from the MNIST training dataset, c-EB was applied to the top-down attentional process on the test pairs and driven by one of the four goals to excite only the pixels relevant to the goal digit.

Figure 3.2 shows the two noisy test pairs and their c-EB generated attention maps according to each goal. c-EB driven by a goal went through the backward pass and excited the pixel

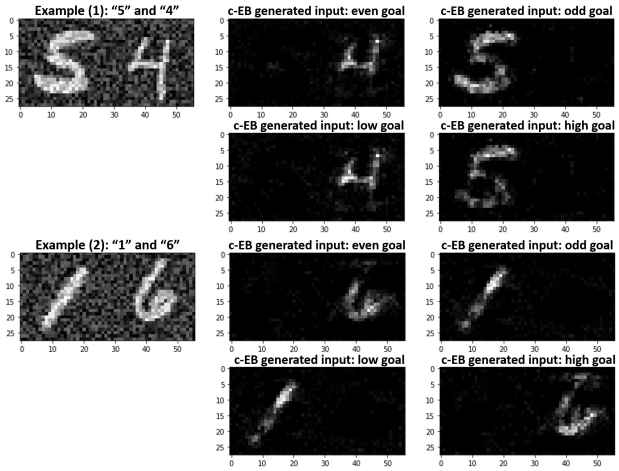


Figure 3.2: Two example test pairs of noisy MNIST digits and their c-EB highlighted results. The two digits in each test pair had the opposite goals both in the parity (even/odd) goal class and in the magnitude (low/high) goal class. These restrictions were not applied to the training pairs during the experiments.

neurons only related to the goal digit. On the irrelevant digit side, most pixel neurons were inhibited instead. Furthermore, background noise on both sides were ignored. Therefore, the goal digits and goal identity neurons could all be predicted correctly with high certainty in the end of the forward pass in these examples. However, even if two goal identities targeted the same goal digit, their highlighted pixels in c-EB visualization were not all the same. It is reasonable because our model had different output heads for the different goal classes.

With the first noisy test pair of “5” and “4” in Figure 3.2 as an example, if an even goal was selected, then both even goal neurons in the left and right digit sides were activated and all other goal neurons were inhibited in the top layer of the backward pass. After the c-EB process that sent contrastive signals through the third hidden layer and added them up for normal EB via the second and first hidden layers to reach the bottom layer (see Figure 3.1), pixels related to the digit “4” were highlighted, whereas pixels related to the distractor digit “5” and the background noise were inhibited.

Training and Testing Process

The training process consisted of incremental learning on noisy MNIST pairs. The original MNIST dataset was split into 60000 digit images for training and 10000 digit images for testing (LeCun et al., 1998). At each training step, 256 pairs of noisy MNIST digits were randomly selected and modified from the original MNIST training set. Every 200 training steps, 2000 noisy MNIST pairs, randomly selected and modified from the original MNIST test set, were used for validation to evaluate the current training progress. The training process stopped after 4400 steps, when the validation accuracy was the highest. Therefore, a total of 1,126,400 noisy MNIST pairs were used for training. The following test process consisted of 10000 pairs randomly selected and modified from the original MNIST test set. Given the sizes of the original MNIST training and test sets, there must be digit overlap within some training pairs or within some test pairs. However, there was no digit overlap between training and test. The two digits in each training pair could have the same or opposite parity (even/odd) and the same or opposite magnitude (low/high), whereas those in each test pair all had the opposite parity and the opposite high and low values.

During training, a log-softmax function, followed by a negative log likelihood function, was applied after the forward pass to the neurons in the output layers that represented a digit, even parity, odd parity, low value, or high value. Then the sum of loss was used to calculate the gradient for each parameter in the model. At the end of each training step, a parameter update was performed based on the current gradient calculated using the Adam optimizer (Kingma and Ba, 2014) with a learning rate of 0.001.

During testing, c-EB drove goal-driven perception by increasing the activity of input neurons corresponding to the goal digit and masking out the neurons corresponding to the distractor digit. In the top layer of our network (Figures 3.1 and 3.4), either two out of the four parity neurons or two out of the four magnitude neurons were activated, depending on the selected

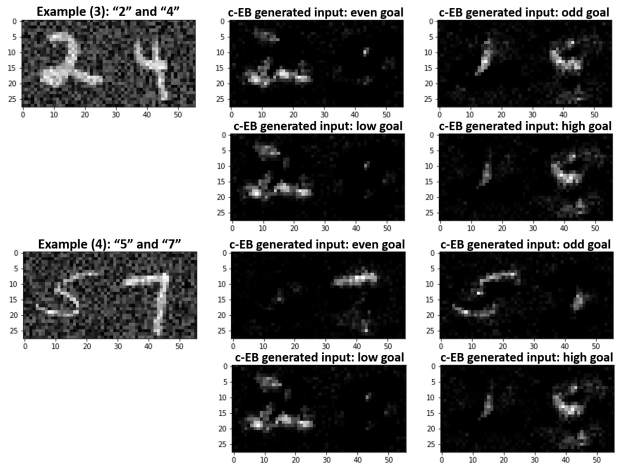


Figure 3.3: Two more example test pairs of noisy MNIST digits and their c-EB highlighted results. The two digits in each test pair had the same goals both in the parity (even/odd) goal class and in the magnitude (low/high) goal class. The tested condition with same parity and/or high/low was not included in later experiments.

goal. For example, if an “odd” goal was selected, the odd neuron for the left digit and the odd neuron for the right digit were both excited, whereas all other goal neurons for both digits were inhibited. This resulted in c-EB in the backward pass to increase attention to this goal and its corresponding digit, and to ignore all other goals (see blue arrows for a parity goal and red arrows for a magnitude goal in Figure 3.1). Such c-EB generated input, with pixels highlighted for the goal digit (Figure 3.2), then went through the forward pass again in a way similar to the training process. However, according to the goal identity, only the parallel hidden layer related to the target goal class was used to predict the goal digit.

If a test pair had same parity or same high or low values (see Figure 3.3), an existing goal would drive c-EB to highlight pixels from both digits. As expected, it shows that this ambiguous situation would cause confusion in the attention system. We assume it would cause confusion and random selection by a human faced with the same stimuli. Thus, we did not present same parity or same high or low values as test pairs.

After this training and testing procedure, the parameters of the fully trained model were fixed for the neuromodulated goal-driven perception experiments.

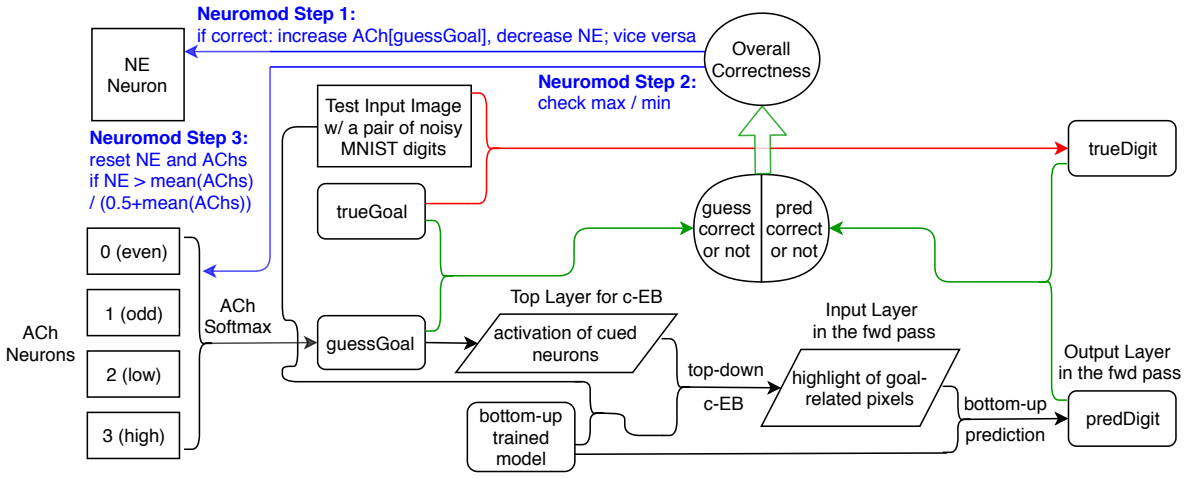


Figure 3.4: The neuromodulated procedure of making digit prediction from a guessed goal for the noisy MNIST-pair experiment.

3.2.2 Neuromodulated Goal-driven Perception

The overall neuromodulated procedure of goal-driven perception is shown in Figure 3.4. As described in Section 3.2.1, the network was trained with pairs of noisy MNIST digits to learn the digits and their parity (even/odd-value) and magnitude (low/high-value) goal classes. Then it was tested by selecting one of the even, odd, low-value, and high-value goals to trigger c-EB in the backward pass and generate an attention map, which further led to prediction of the digit and goal in the succeeding forward pass. After the robustness of c-EB prediction was verified, we applied ACh and NE neuromodulatory neurons to track the expected and unexpected uncertainties respectively and guess the goal for each trial. The guessed goal was applied to the top layer for c-EB as the intended goal for the current test pair. The guessed goal and predicted digit were compared with the true goal and true goal-related digit. The prediction was used to modify the neuromodulatory activities for the next trial, as will be described in Section 3.2.2.

ACh and NE Neuromodulation

For goal-driven perception, the network must select a goal when they are uncertain and unknown *a priori*. Similar to a model of the ACh and NE neuromodulatory systems proposed by Yu and Dayan (2005), the goal target (even, odd, low, or high value) was rewarded with a probability (goal validity), but that goal would change periodically (goal identity). ACh neurons tracked *expected uncertainties* of the potential goals. NE neurons tracked *unexpected uncertainties*, and responded phasicially when a goal identity change is detected. When the NE system responded phasically, it caused a network reset by re-initializing the ACh and NE neural activities, which allowed rapid adaptation under novel conditions.

Algorithm 1 shows the logic of our ACh and NE neuromodulatory model. There were $K = 4$ ACh neurons, each neuron corresponding to a goal (i.e., even, odd, low, or high value), and one NE neuron. One of the four attentional goal tasks was selected as the major goal for each goal switch. The true goal identity in each trial was set to either the major goal or the minor goal according to the goal validity (see Section 3.2.2 for details). The true goal digit was obtained from the labels of the test pair by using the true goal identity. The activities of ACh neurons were input to a softmax function for goal selection:

$$p(goal)_i = \frac{\exp(\beta \cdot ACh_i)}{\sum_{j=1}^{K} \exp(\beta \cdot ACh_j)} \text{ for } i = 1, 2, ..., K, \tag{3.4}$$

where β is the temperature governing exploration versus exploitation and $p(goal)_i$ is the probability of selecting goal i. This guessed goal activated two neurons related to the goal in the top layer of our network architecture (Figures 3.1, 3.4), which directed c-EB in the backward pass to activate the goal-relevant pixels in the test pair and then predicted the digit in the forward pass. If the prediction was correct (which means that the guessed goal identity matched the true goal identity and the predicted digit matched the true goal digit), the ACh level corresponding to the guessed goal (i.e., ACh_g) increased and the NE level

decreased; the opposite would happen otherwise:

$$(ACh_g)_t = \begin{cases} \min(ch_{correct}(ACh_g)_{t-1}, ch^{max}) & \text{if correct,} \\ \max(ch_{wrong}(ACh_g)_{t-1}, ch^{min}) & \text{otherwise,} \end{cases} \tag{3.5}$$

$$NE_t = \begin{cases} \max(ne_{correct} \cdot NE_{t-1},\ ne^{min}) & \text{if correct,} \\ \min(ne_{wrong} \cdot NE_{t-1},\ ne^{max}) & \text{otherwise,} \end{cases} \tag{3.6}$$

where $[ch^{min}, ch^{max}]$ and $[ne^{min}, ne^{max}]$ were ranges for ACh and NE levels. $ch_{correct}$ and ne_{wrong} must be set within $[1.0, 2.0)$, and ch_{wrong} and $ne_{correct}$ must be within $(0, 1.0]$. If the NE level was above a threshold θ^{reset}, ACh and NE activities were reset to baseline levels (Yu and Dayan, 2005):

$$\theta^{reset} = \frac{\left(\sum_{i=1}^{K} ACh_i\right)/K}{0.5 + \left(\sum_{i=1}^{K} ACh_i\right)/K}. \tag{3.7}$$

Our settings for constant parameters of the neuromodulation process are listed in Algorithm 1. However, there was a wide range of parameter values that could be used to produce stable results. The randomness in Algorithm 1 followed a uniform distribution within the ranges specified, except that selecting *guessGoal* required the softmax distribution.

Goal Selection

We added an online neuromodulatory model (Figure 3.4 and Algorithm 1) to the head of the network architecture in the backward pass to regulate goal selection automatically. In these experiments, the goal (with goal identities of even, odd, low, or high value) had to be learned from experience. It might be noisy and rewarded with some probability (i.e., goal

Table 3.1: Relationship Between Goal Actions and Object Labels.

Action	Role	Objects in Role
eat	obj	banana, apple, sandwich, orange, donut, carrot, broccoli, hot dog, pizza, cake
	instr	fork, knife, spoon, bowl, cup
work	instr	laptop, tv, mouse, keyboard
read	obj	book
	instr	laptop, cell phone
say-hi	obj	person

validity).

Automatic goal selection was tested in 10 runs to measure the average performance. In each run, one of the four attentional goal tasks was randomly selected as the major goal, which stayed the same every 400±30 trials for 10 switches. The minor goal identity came from the same goal class as the major goal identity. For example, if the major goal was "high", then the minor goal became "low" in the same magnitude goal class; or if the major goal was "even", then the minor goal became "odd" in the same parity magnitude class. The true goal identity was set to either the major goal or the minor goal randomly according to the validity distribution per trial. The true goal digit was obtained from the labels of the test pair of noisy MNIST digits using the true goal identity.

The goal validity values (i.e., 0.99, 0.85, and 0.70) were chosen to correspond with Yu and Dayan (2005). The major goal validity was randomly chosen among the three values each time the major goal identity got switched in a run. The minor goal validity was (1− major goal validity).

3.2.3 Action-Based Attention in a Robot Experiment

To test whether the goal-driven perception model could generalize to a more real-world application, we tested the model in an action-based attention task on the Toyota Human

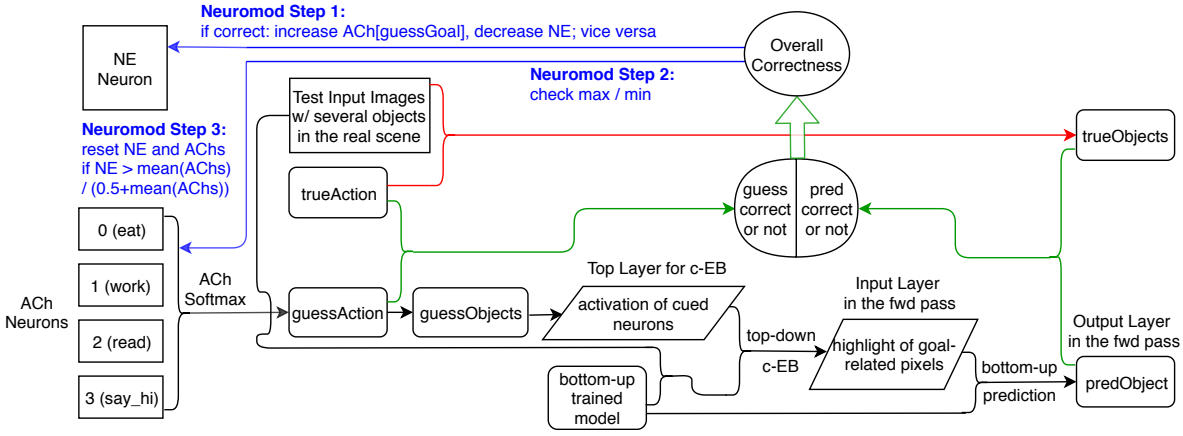


Figure 3.5: The neuromodulated procedure of making object prediction from a guessed goal action for the indoor robot experiment.

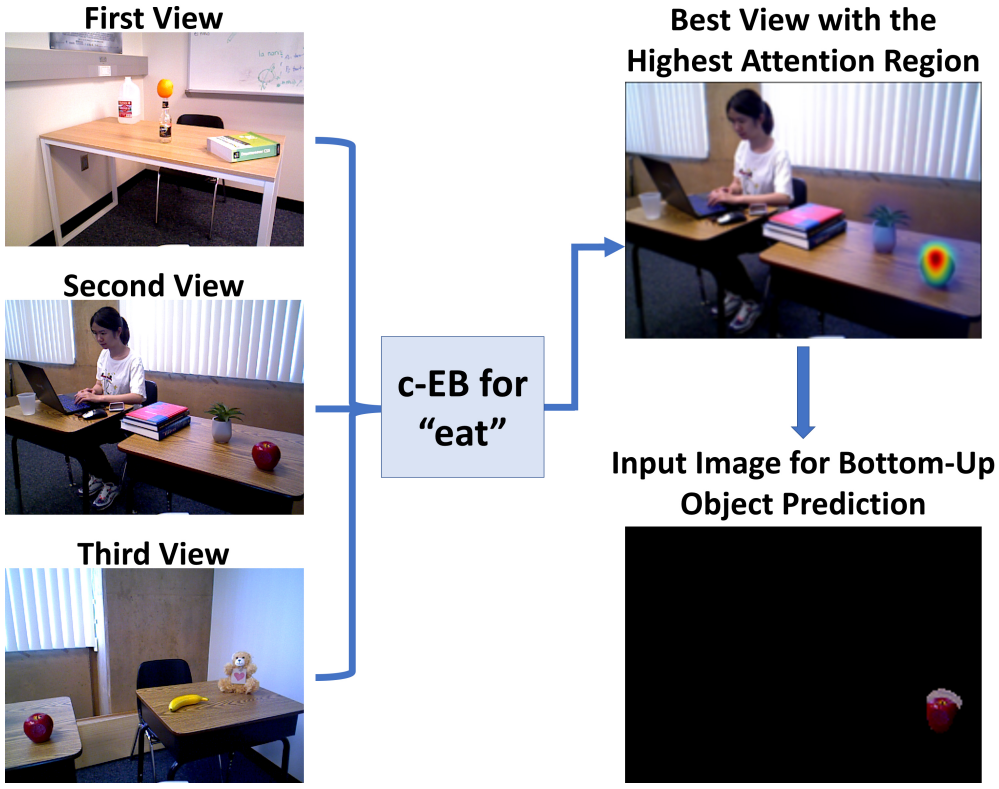


Figure 3.6: The top-down attentional search process for a guessed action "eat" based on three different real indoor views to select the highest attention region for bottom-up object prediction.

Figure 3.7: The test scenario for the indoor robot experiment.

Support Robot (HSR). For the second experiment, we had four goal actions (i.e., "eat", "work-on-computer", "read", "say-hi") that were associated with images of objects seen by the HSR. Given a desired action, the task for the HSR was to guess the action and direct attention to the object in the scene that could achieve that action. For example, the action "eat" might result in attention to an "apple".

For object classification, we used the Microsoft COCO dataset (Lin et al., 2014) to train a GoogLeNet (Szegedy et al., 2015) via the Caffe framework (Jia et al., 2014) instead of the MNIST-pair network shown in Figure 3.1. An advantage of the COCO dataset was it used segmentation to localize individual object instances in the image, which was more accurate and more helpful for top-down attention than using bounding boxes.

For each run of this experiment, desired actions were randomly switched every 50 trials for 10 switches. In each trial, each image from three capture angles within an indoor scene was loaded as the input to the pretrained bottom-up model and went through a forward pass with the output layer specified as "loss3/classifier" in Caffe. The number of output prediction classes was set to 80, same as the number of object labels available for the COCO dataset (Lin et al., 2014). Then the c-EB method was applied for the top-down attention process. As in the MNIST-pair experiment, the NE-ACh neuromodulation process with softmax on the ACh activities was applied for goal (action) selection (see Figure 3.5).

Table 3.1 linked the guessed action with all related objects – from 80 COCO labels regardless of their semantic roles – which might or might not exist in the test scenario. Those activated object neurons in the top layer drove the c-EB through the second top layer "pool5/7x7_s1" and then normal EB to the bottom layer "pool3/3x3_s2" to generate attention maps related to this guessed goal action. The notation "pool5/7x7_s1" referred to a pooling layer with a kernel size of 7x7 and stride of 1. Only the highest attention region (with normalized attentional strength above the threshold of 0.1) corresponding to one of the three real captures maintained its original pixel values, whereas all other parts of the image became black (see Figure 3.6). This attention-modulated image became the input to the forward pass of the network and generated object prediction via the top layer. Three conditions needed to be satisfied to generate a overall correct match for that trial: (1) the guessed action matched the true action; (2) the predicted object matched the real object in the scene; (3) the predicted object was associated with the guessed action.

The test environment for this experiment was a classroom scenario as shown in Figure 3.7. The test agent was a Toyota HSR (Yamamoto et al., 2018). During each trial, the HSR first guessed an action using the activity of the neuromodulatory neurons and linked it with objects using the semantic network. The HSR then moved from a starting point to the center of the testing scenario, where it captured three images from different view angles using the RGB-D camera. After the attention network predicted an object as described above, the HSR moved towards the object and either picked up the object if it was grabbable (e.g., apple) or pointed at the object with its arm (e.g., laptop). At the trial end, the HSR would present the object to a user who would respond with a "YES" if it the object matched his/her desired action or otherwise with a "NO". This feedback would be used by the neuromodulatory model to adjust the activity levels of both the NE neuron and the ACh neuron related to the current guessed action before guessing the action for the next trial.

Table 3.2: Prediction for 10,000 test pairs of noisy MNIST digits.

Goal Task	% Correct Digit Prediction	% Correct Goal Prediction
Even	92.03	99.50
Odd	91.15	99.75
Low	95.39	99.54
High	87.46	98.22

3.3 Results

Section 3.3.1 shows how the network attends to correct goals and predicts the digits that correspond to those goals. Section 3.3.2 demonstrates the ability of the neuromodulatory head to learn goals based on its experience in uncertain domains. Section 3.3.3 shows the necessity of having both NE and ACh neurons to correctly predict goals. Section 3.3.4 compares the performance of our method with two benchmarks. The experiments in these sections are carried out with noisy MNIST pairs. Section 3.3.5 shows how our goal-driven perception method generalizes to an action-based attention task with a physical robot.

3.3.1 Digit Prediction with c-EB and Noisy MNIST Pairs

The training process was carried out for 4,400 steps, including 256 noisy MNIST pairs modified from the original MNIST training set per step. The prediction performance of the fully trained model was tested on 10,000 pairs of noisy MNIST digits modified from the original MNIST test set (LeCun et al., 1998). Table 3.2 shows the digit and goal prediction results with c-EB driven by one of the four goal tasks (i.e., even, odd, low, or high value).

The goal was predicted along with the digit in the output layers for each forward pass. As shown in Table 3.2, the model predicted the goal correctly over 99% of the time, meaning that after the backward and forward pass the most active neuron predicting the goal

matched the true goal. The model predicted the goal digit correctly over 90% of the time, meaning that the most active digit neuron after the backward and forward pass matched the expected digit based on the goal (see Table 3.2). This indicates that the goal tasks were successfully understood by the c-EB process to highlight related pixels. Although the statistics of the high-value goal task was slightly weaker than that of the other three goal tasks, the performance was still robust overall.

In the next section, we show how this network can autonomously predict goals in uncertain domains.

3.3.2 Goal-Driven Perception with Uncertainties

The robust digit and goal prediction results using c-EB (see Section 3.3.1) assured that the network architecture could be applied to situations where goals uncertain and contexts are unknown. Therefore, the next step was to test the reliability and flexibility of our proposed neuromodulation model for predicting goals in a noisy, dynamic environment.

Figure 3.8 shows typical runs of our neuromodulated system for three major validity settings. For each major validity of 0.99 (Figure 3.8a), 0.85 (Figure 3.8b), or 0.70 (Figure 3.8c). Within each sub-figure (a), (b), or (c), the first subplot includes the true goals (labeled as "major goal" and "minor goal") and ACh-guessed goals (labeled as "guess"); the second and third subplots show NE and ACh levels. Note that the ACh neuron corresponding to a major goal quickly increased driving attention to the most likely goal, as well as suppressing attention to distractors. In cases where the major goal validity was low, the ACh neuron corresponding to the minor goal was also activated, resulting in more exploration and a higher chance of guessing the minor goal. Interestingly, the prediction during exploration tended to remain in the same goal class. When there was a change in the goal identity, the NE neuron quickly recognized the change and responded with spike of activity. This caused the activities in the

Table 3.3: Average goal-driven perception performance on noisy MNIST pairs over 10 runs for each of the four goal validity settings. The first three rows of data correspond with the first experiment of one major goal validity. The last row relates to the second experiment of randomly switched goal validity. *p_valid* means the major goal validity, and $(1 - p\_valid)$ means the minor goal validity. In each run, the major goal was randomly picked every 400±30 trials for 10 switches. The minor goal was selected from the same goal class. The β value for the softmax function (see Equation 3.4) was set to 0.7.

Major Goal Validity	Minor Goal Validity	% Correct Major Goal	% Correct Minor Goal	% Incorrect ACh Softmax Goal Guessing	% Incorrect c-EB Digit Prediction	Lag Length (trials)
0.99	0.01	86.1	0.0	7.8	6.1	21
0.85	0.15	73.0	0.3	20.4	6.3	29
0.70	0.30	57.9	1.5	34.3	6.3	48
p_valid	1-p_valid	75.1	0.7	18.0	6.2	30

goal prediction network to reset, and a short period of exploration before the system found the new goal identity. Lower major goal validity led to longer exploration, especially after a goal identity switch, as well as more frequent NE bursts.

We also ran experiments where the goal validity could change during the run. Figure 3.9 shows the performance of a typical run with random switching among three major goal validity options, 0.99, 0.85, and 0.70. Similar to Figure 3.8, the system in this setting still focused more on the major goal. With lower major goal validity (i.e., when the major goal appeared less frequently, see 0.70 in Figure 3.9), the NE neuron fired phasically more frequently; meanwhile, the activity level of the major goal's ACh neuron oscillated more frequently with larger amplitude, giving higher potential for the minor goal's ACh neuron to fire at a low level. Because both the goal validity and goal identity changed during a run, the exploration period lasted longer with a lower major goal validity of 0.85 or 0.70. However, this also led to higher prediction accuracy of the minor goal.

Table 3.3 shows the performance of goal and digit prediction on the noisy MNIST pairs over 10 runs for each validity setting. The third and fourth columns refer to the percent of trials at which the goal digit prediction was correct. The fifth column refers to the percentage

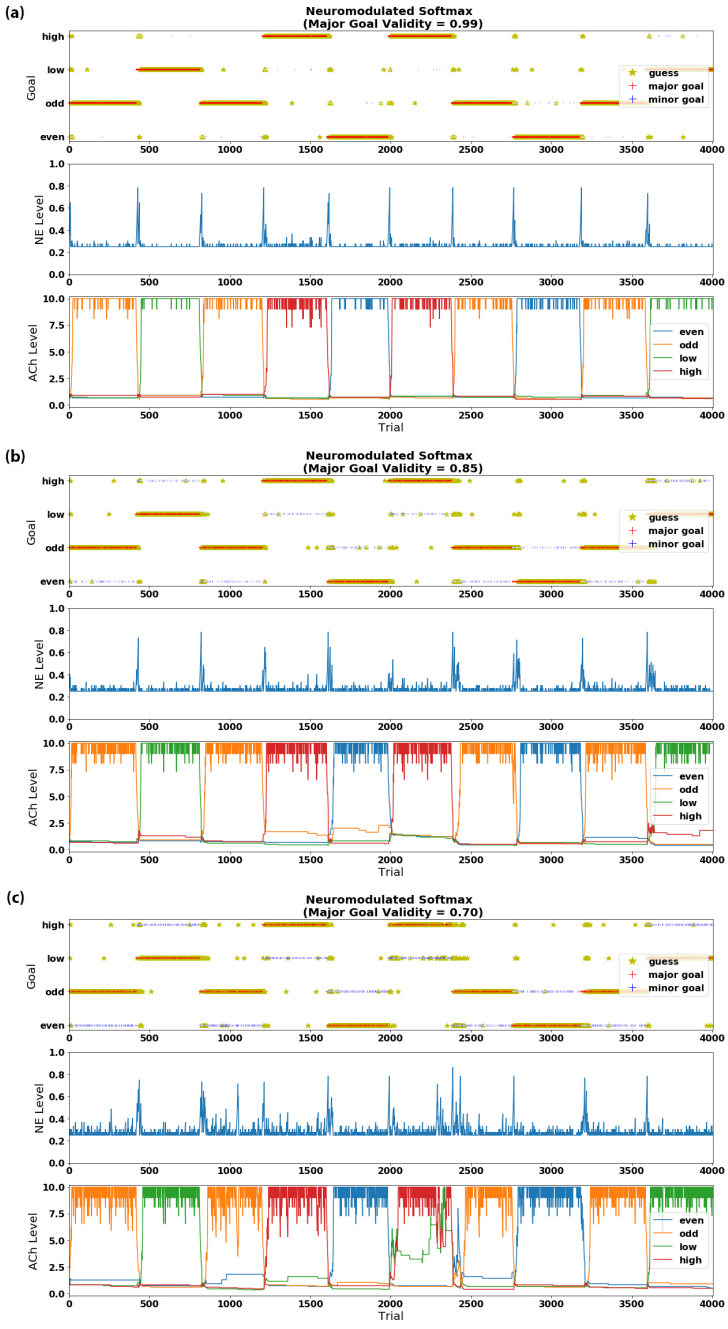


Figure 3.8: Visualization of goal-driven perception performance on noisy MNIST pairs with the major goal validity chosen from (a) 0.99, (b) 0.85, or (c) 0.70. The major goal identity was randomly picked every 400±30 trials for 10 switches in a run. The minor goal was the other goal in the same class of the major goal. For example, if the major goal "odd" had validity of 0.70, the minor goal "even" had validity of 0.30 until the next major goal switch. For each sub-figure, the top subplot shows guessed goal identities (in yellow) and true goal identities (either major goals in red or minor goals in blue); the middle and bottom subplots show NE and ACh levels. A softmax function (see Equation 3.4, with $\beta = 0.7$) was applied to ACh levels for goal guessing. See text for details.

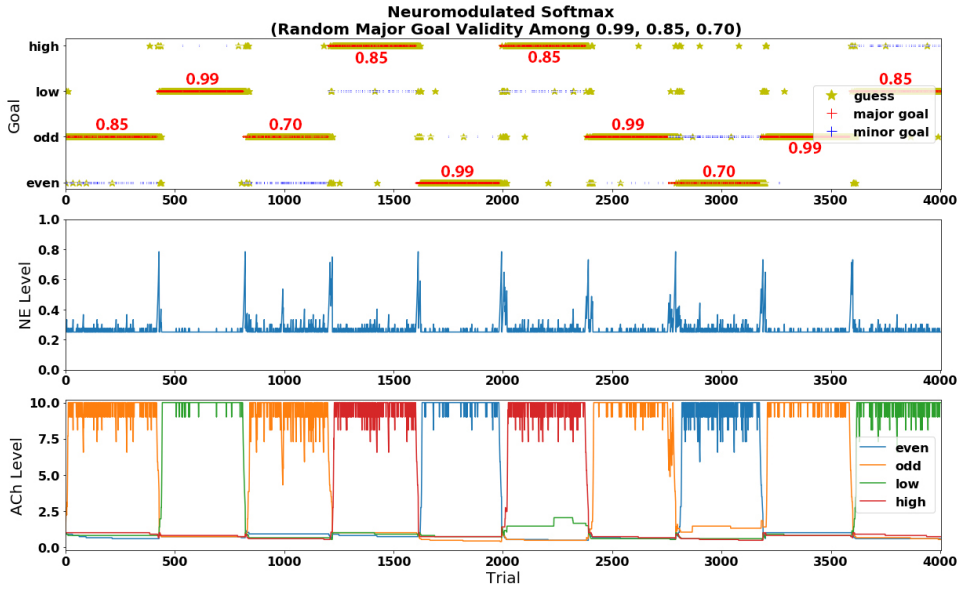


Figure 3.9: Visualization of goal-driven perception performance on noisy MNIST pairs with the major goal validity randomly switching among 0.99, 0.85, and 0.70. All other settings were the same as shown in Figure 3.8.

of incorrect goal guessing based on the ACh softmax distribution (see Equation 3.4, with $\beta = 0.7$). The sixth column refers to the percent of incorrect digit predictions with c-EB driven by the guessed goal, when the ACh-guessed goal already matched the true goal. The seventh column refers to lag length of choosing the correct goals, which was computed as the number of trials between the first trial of a major goal switch and when the network started consistently making correct goal prediction 80% of the time over the last 10 trials.) The first three rows provide average statistics for runs at which a single goal validity was tested (see also Figure 3.8), and the last row corresponds with runs at which the goal validity could change randomly among three options during a run (see also Figure 3.9).

3.3.3 Ablation Studies

We wanted to understand the effect of each neuromodulator on the network's ability to select goals. Therefore, we simulated ablation studies on ACh and/or NE neurons in a randomly changing goal validity experiment. These ablations had drastic effects on performance (see

Table 3.4: Average goal-driven perception performance on noisy MNIST pairs over 10 runs for each of the four ablation conditions on the NE and/or ACh neuron(s). In each run, the major goal was randomly picked among the four goal options every 400±30 trials for 10 switches. For each major goal switch, the major goal validity was selected randomly among 0.99, 0.85, and 0.70. The minor goal was selected from the same goal class. The β value for the softmax function (see Equation 3.4) was set to 0.7.

Ablated Neuron(s)	% Correct Major Goal	% Correct Minor Goal	% Incorrect ACh Softmax Goal Guessing	% Incorrect c-EB Digit Prediction	Lag Length (trials)
None	75.1	0.7	18.0	6.2	30
NE	70.8	1.1	21.6	6.5	54
ACh	19.8	2.9	70.9	6.4	400
NE & ACh	19.9	2.9	70.9	6.3	400

Table 3.4 and Figure 3.10) compared to the complete network. With ablation of the NE neuron (see Figure 3.10a), there was no scheme for network reset. The ACh neurons were still able to track the major goal switches. However as time elapsed, it took longer for the ACh activity level corresponding to the major goal to rise significantly and properly after goal switches, as measured by the lag length. With ablation of the ACh neurons (see Figure 3.10b), the goal guessing became random. The firing rate of the NE neuron increased rapidly in the beginning and stayed at extremely high values afterwards. With ablation of both ACh and NE neurons (see Figure 3.10c), there was no firing activity of either the NE or ACh neuron(s), and thus the goal guessing was random. These ablation studies demonstrate the necessity of having one system track the expected uncertainties (ACh) of goals and another respond appropriately when the goal distribution changes (NE).

3.3.4 Goal Selection Method Comparison

In the neuromodulated procedure of our model (see Figure 3.4), the goal was selected by calculating the softmax distribution based on the activities of the four ACh neurons (see Equation 3.4, with $\beta = 0.7$). The softmax function was important for raising the chance of

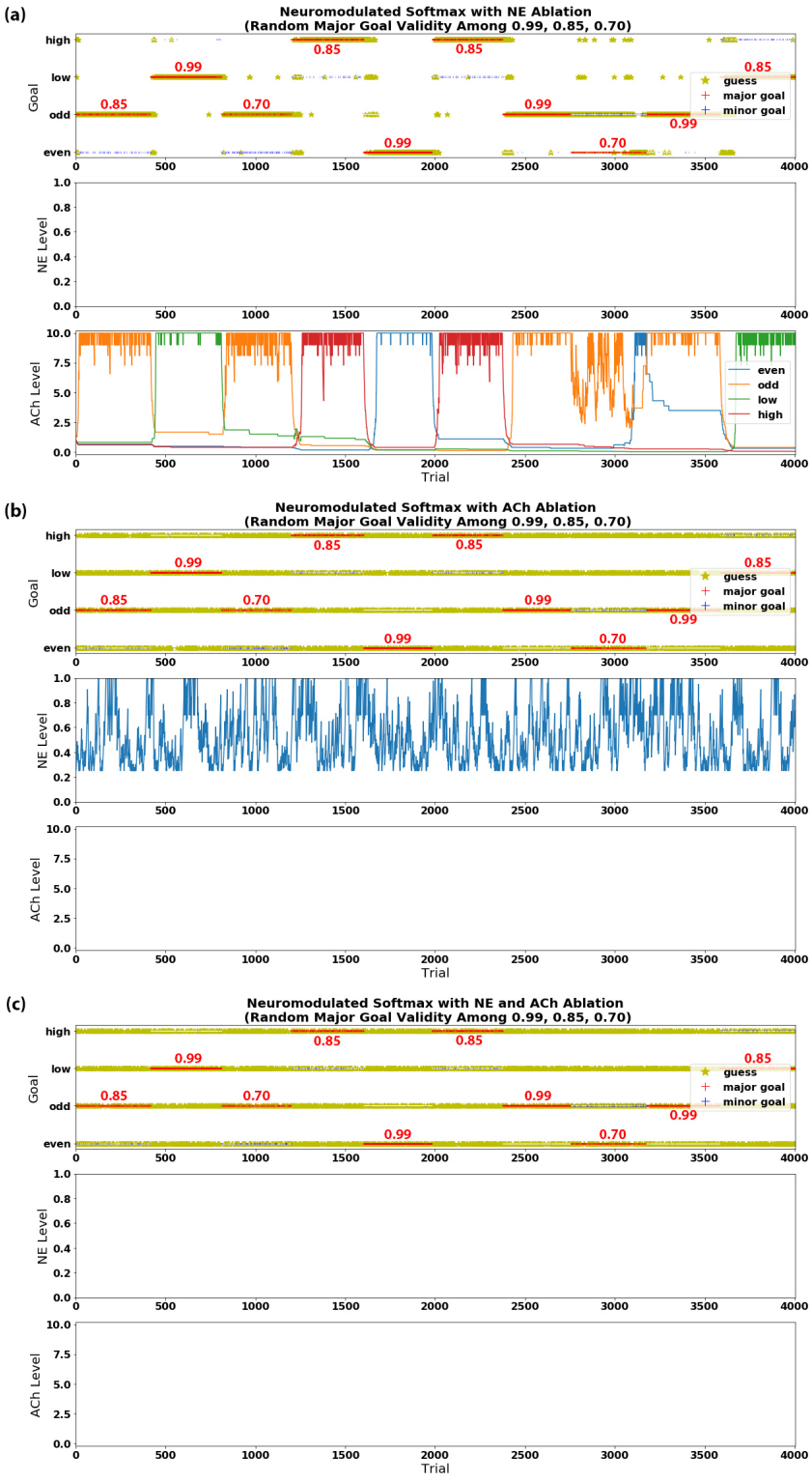


Figure 3.10: Visualization of goal-driven perception performance on noisy MNIST pairs with the major goal validity randomly switching among 0.99, 0.85, and 0.70, after (a) NE ablation, (b) ACh ablation, and (c) NE and ACh ablation. All other settings were the same as shown in Figures 3.8 and 3.9.

choosing the minor goal when the major goal validity was low (e.g., 70%). We compared softmax to a winner-take-all (WTA) selection method, which still used the neuromodulatory head.

In addition, we compared the neuromodulatory head to another benchmark, which we call "random-or-fixed". In the "random-or-fixed" benchmark, a predicted goal was randomly selected until it matched the true goal. Then the goal selection was fixed until a mismatch appeared. In other words, whether the guessed goal was random or stayed the same depended on whether there was a mismatch or match in the goal guessing process of the previous trial. Table 3.5 shows the performance comparison among neuromodulated softmax (shown in Figure 3.9), neuromodulated WTA (shown in Figure 3.11), and "random-or-fixed" (shown in Figure 3.12) on the noisy MNIST pairs. All three methods had similar lag lengths. Although the "random-or-fixed" method generated the highest percentage of minor goal matches, it was mostly caused by random guesses among all four goals, which also lowered the percentage of major goal matches. Therefore, the neuromodulation process was important for quickly following the desired goal class without hesitating over all four goals after each major goal switch. For neuromodulated softmax and neuromodulated WTA, their overall accuracy of major and minor goal guessing was quite similar. However, comparing Figure 3.9 for softmax with Figure 3.11 for WTA, we observed that the softmax function allowed higher chances of selecting the minor goal during the intervals at which the major goal validity was low, whereas the WTA function would like to select the major goal regardless of its true validity and could cause a much longer lag when the major goal validity dropped.

3.3.5 Goal-Driven Perception on Robot

To demonstrate that our model could generalize to a more practical application than MNIST digits, we tested our model on a human support robot that had to guess an action with the

Table 3.5: Average goal-driven perception performance on noisy MNIST pairs over 10 runs among neuromodulated softmax (with $\beta = 0.7$), neuromodulated WTA, and "random-or-fixed". In each run, the major goal was randomly picked among the four goal options every 400±30 trials for 10 switches. For each major goal switch, the major goal validity was selected randomly among 0.99, 0.85, and 0.70. The minor goal was selected from the same goal class.

Goal-Guessing Method	% Correct Major Goal	% Correct Minor Goal	% Incorrect Goal Guessing	% Incorrect c-EB Digit Prediction	Lag Length (trials)
Neuromodulated Softmax	75.1	0.7	18.0	6.2	30
Neuromodulated WTA	76.4	0.8	16.5	6.3	24
"Random-Or-Fixed"	63.1	1.7	29.0	6.2	23

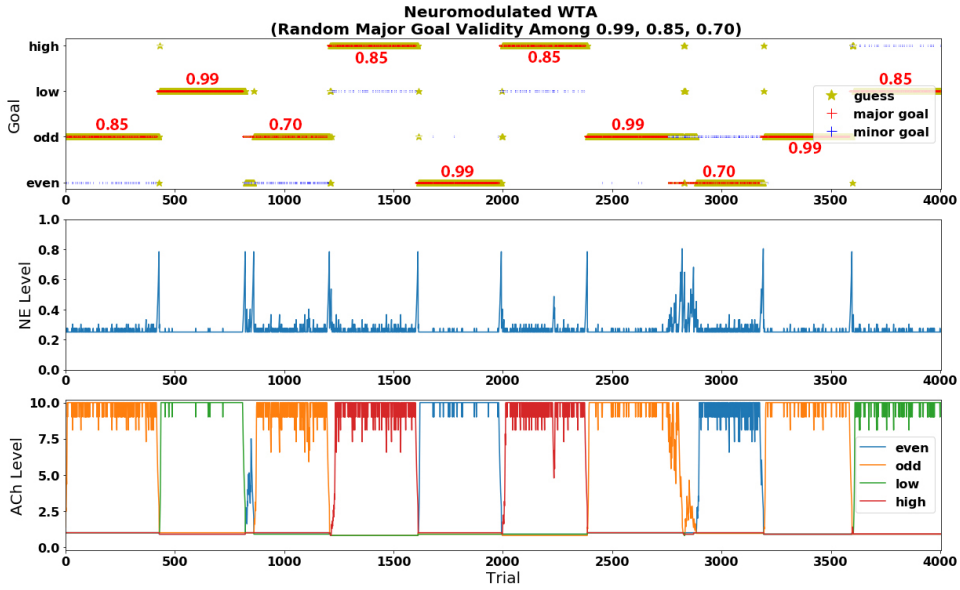


Figure 3.11: Visualization of goal-driven perception performance on noisy MNIST pairs with the major goal validity randomly switching among 0.99, 0.85, and 0.70. In this neuromodulated benchmark, WTA replaced the softmax distribution in our model for cholinergic goal guessing.

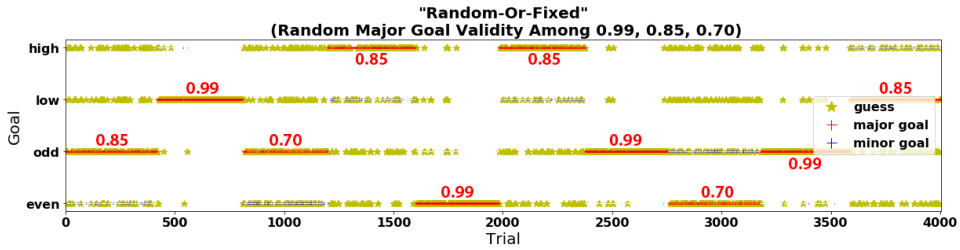


Figure 3.12: Visualization of goal-driven perception performance on noisy MNIST pairs with the major goal validity randomly switching among 0.99, 0.85, and 0.70. In this "random-or-fixed" benchmark, whether the guessed goal was random or stayed the same depended on whether there was a mismatch or match in the goal guessing process of the previous trial.

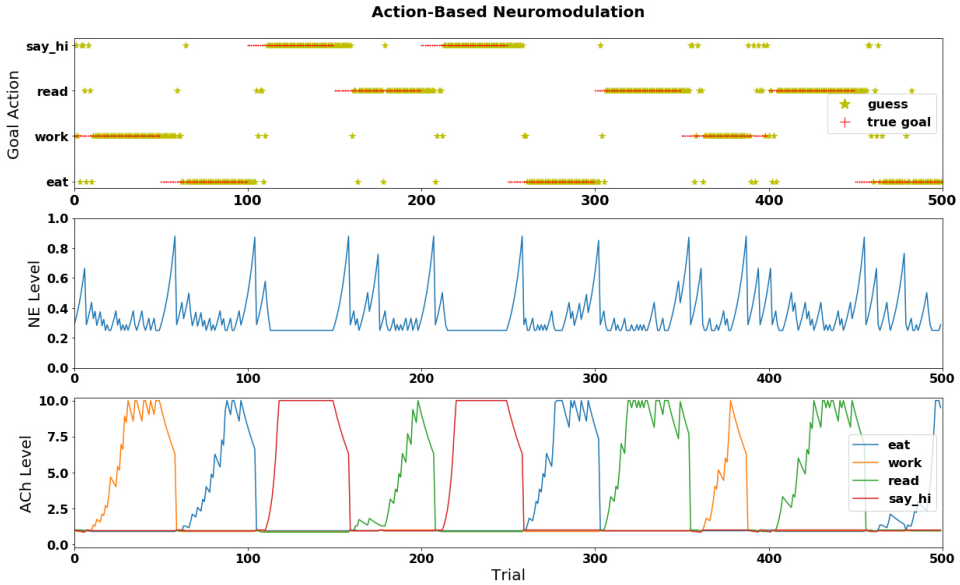


Figure 3.13: Visualization of action-based goal-driven perception performance on different angles of robot views in an indoor scenario. The true goal action was randomly picked every 50 trials for 10 switches in a run. A softmax function (see Equation 3.4, with $\beta = 10$) was applied to ACh levels for action guessing. See text for details.

neuromodulatory head, selectively attend to an object that corresponds to that action, and retrieve the object. The methods for this scenario were described in Section 3.2.3.

Figure 3.13 shows the performance of a typical run of the action-based goal-driven perception. Its neuromodulation process was similar to Algorithm 1 for the noisy MNIST-pair experiment. Changes included using input images from three view angles of the HSR's camera, a fixed goal (action) validity of 1, and several parameter value adjustments (i.e., $\beta = 10$, $trial\_interval = 50$, $trial\_range = 0$, $ne_{correct} = 0.75$, $ne_{wrong} = 1.15$, $ch_{correct} = 1.35$, $ch_{wrong} = 0.95$). The average goal selection error for 5 runs was 23.8%, which was higher than the noisy MNIST-pair experiment (see Table 3.3) because of shortened trial interval for each goal (action) switch. The average c-EB object prediction incorrectness was 30.6%. The average lag length was 13 trials. Uncertainties in each trial were addressed by possible object location switch, possible object removal and/or introduction, possible multi-instances of the same object(s), and slight view angle adjustment, in addition to possible true action switch (i.e., every 50 trials, not given to the agent). A complete trial with HSR in the testing room can be seen in the YouTube video (https://youtu.be/DUy-0fDZEvY).

3.4 Discussion

3.4.1 Main Findings

In this chapter, we showed that a neuromodulated goal-driven perception model, which combines ideas from neuroscience with goal-driven perception in machine learning and artificial neural networks, could track context and flexibly shift attention to intended goals. Among many top-down attentional systems, we adapted c-EB (Zhang et al., 2018) as part of our model because of its similarities to how the ACh neuromodulatory system both increments attention to a goal and decrements attention to a distractor (Oros et al., 2014; Baxter and

Chiba, 1999). Goals are often unknown and need to be discovered. The c-EB algorithm was modified to support multiple goals. After training, the biologically inspired algorithm could rapidly learn the context without supervision, flexibly apply attention to the appropriate goal, and rapidly detect and re-adapt to context changes.

Neural Implementation of Uncertainty Tracking

Yu and Dayan (2005) proposed a Bayesian model of neuromodulation in which the ACh system tracked expected uncertainty and the NE system tracked unexpected uncertainty. The present chapter advances this work in two ways to support goal-driven perception: 1) The Bayesian model was recast as a neural model to make it compatible with neural networks. The neuromodulators were implemented as a neural network layer to drive attention toward a goal digit and divert attention away from distractors. 2) A neural network reset was implemented to rapidly re-adapt when a goal changes.

Neuroanatomical studies show the basal forebrain, which contains ACh neurons, has topographical connections specific to stimulus modalities and values (Zaborszky, 2002). Therefore, different ACh neurons tracked the expected uncertainties of different potential goals. In a dynamic situation, the goal identity can change unexpectedly. Empirical evidence suggests that the NE system detects such changes and generates a "reset" signal to discard prior expectations when these expectations are violated (Bouret and Sara, 2005; Grella et al., 2019). In our experiments, the NE system rapidly recognized a change in the goal contingency, and drove a reset of ACh and NE activities. This caused the neural network to quickly explore new goals. It should be noted that the "reset" does not erase the learned object categories (e.g., digit parity and magnitude). Instead, it clears the prior likelihood of potential goals, and results in a rapid re-adaptation to the new goal distribution.

In the real world, goals are often uncertain and unknown. In our noisy MNIST-pair experi-

ment, the goal validity (i.e., probability of a goal being rewarded) ranged from 0.99 to 0.85 to 0.70, and the system had to respond by either choosing the most likely goal or exploring alternative goals. Furthermore, the experimental design had a hierarchy of goals. For example, the goal would be to attend to the parity goal class and the sub-goal might be to reward odd digits 70% of the time and even digits 30% of the time. Interestingly, the neural network would often stay within a goal class (i.e., to choose parity and not magnitude).

In the robot action-based attention experiment, the objects linked with a predicted goal action might or might not exist in the views and may probably be at different locations. The adapted c-EB attention mechanism could pay significantly higher attention to existing objects. Selecting the highest attention region helped further with object localization and prediction. In both the MNIST-pair and real-scenario experiments, the unexpected major goal (action) switch after some trials could be quickly caught by the network within an acceptable lag.

Exploration and Uncertainty Seeking

Exploring options, rather than always choosing the most likely goal, is known as probability matching behavior (Wozny et al., 2010). Similar to the results presented here, humans tend to underselect the most rewarding goal (Craig et al., 2016). Such behavior may be due to feature exploration, as subjects test hypotheses by switching between the features before deciding upon their most rewarding goal. In rodent studies, it has been shown that rats will seek uncertainty, and that this uncertainty seeking is governed by the ACh system (Naude et al., 2016). These uncertainty seeking strategies that appear in natural systems may be advantageous for artificial systems that are deployed in dynamic environments.

3.4.2 Related Work

Top-down task-driven attention is an important mechanism for efficient visual search in humans and artificial systems (Baluch and Itti, 2011). Many computational models of attention have been proposed and implemented to either explain top-down attention or develop an application inspired by these mechanisms (Tsotsos et al., 2015; Tanner and Itti, 2017, 2019). Of particular interest are attentional systems that can leverage the power of CNNs. In these cases attentional information can propagate backwards, highlighting the features of a given goal (Zhang et al., 2018; Zhou et al., 2016). Similar to the effect of the ACh system to increment attention to a goal and decrement attention to distractors (Baxter and Chiba, 1999), Zhang et al. (2018) proposed an Excitation Backprop (EB) mechanism with a contrastive top-down signal to enhance the perception of goal features. Similarly, Zhou et al. (2016) proposed a technique called Class Activation Mapping (CAM) for identifying regions in an attention map. Selvaraju et al. (2017) proposed Gradient-weighted Class Activation Mapping (Grad-CAM) to highlight regions of interest and generate visual explanations. Their model could be applied to any CNN with no re-training. Similarly, our proposed model can work with any CNN. Moreover, our model replaces the Winner-Take-All mechanism or rigid probabilistic methods, with a flexible and adaptable layer based on neurobiological neuromodulation.

Intrinsic rewards and curiosity seeking have similarities to the exploration due to uncertainty demonstrated by our model. These intrinsic reward systems typically are rewarded for exploring infrequently observed states (Burda et al., 2018; Achiam and Sastry, 2017; Pathak et al., 2017). Whereas the model introduced here selects goals based on the expected uncertainty of stimuli. In future work, it may be of interest to combine intrinsic rewards with uncertainty seeking.

Uniqueness of Our Model

Our model is unique compared to existing attention models, which only focus on highlighting predefined (and pre-trained) goal objects in test images (Cao et al., 2015; Cho et al., 2015), without any ability to deal with unpredictable switching and validity of goals. c-EB, which we adapted in our work, has been shown to achieve top-down attention competitively and robustly (Zhang et al., 2018). Moreover, it is similar to how the ACh neuromodulatory system both increments attention to a goal and decrements attention to a distractor (Oros et al., 2014; Baxter and Chiba, 1999). The neuromodulatory layer on top of a top-down attentional network demonstrates a means toward goal-driven perception where the system can autonomously learn which objects to attend to and which objects to filter out in the noisy, dynamic setting.

In addition to the unique aspects of our neuromodulatory model, its robustness was ascertained via comparisons with neuromodulated WTA and "random-or-fixed" benchmarks. We also used ablation studies to show the necessity of having both ACh and NE neuromodulators to track the expected and unexpected uncertainties of goals and respond appropriately when the goal distribution changes. Further, generalization was ascertained via the HSR implementation in a real indoor scenario.

3.4.3 Future Directions

Handling New Goals

In the present work, the goal classes were known, and the system guessed the appropriate goal given a goal identity and goal validity. However, the system might need to adapt to new goals or new goal classes. Adding multiple heads to the output layer of the network is one way this could be handled. This would not require retraining the stimuli (e.g., digits or real

objects), but some additional training for the new goal classes. However, the architecture might be more scalable with a single head that learns the goals online without any *a priori* assumptions. Similar to the present model, these unknown goals would initially be guessed. After sufficient reward feedback, the model would associate different goals with different reward likelihoods. The introduction of the ACh/NE neuromodulation should make the goal search fast and flexible. This will be explored in future iterations of our work.

Different Attentional Mechanisms

The choice of c-EB for a top-down attentional mechanism was motivated by its similarity to ACh system and its affect on top-down attention. However, as mentioned above, we believe that the proposed system could also work with other state-of-the-art attentional mechanisms, including the CAM (Zhou et al., 2016) and its more general variation Grad-CAM (Selvaraju et al., 2017). As long as the neural network structure can support an additional neuromodulation layer, and there is some means to flow goal information from the top to lower layers, our neuromodulatory goal-driven perception system should be compatible.

Application for Artificial Intelligence (AI)

We had shown the compatibility of the adapted c-EB attention mechanism with the Microsoft COCO dataset and an indoor scenario. Our model is applicable in broader AI scenarios. If a system (e.g., a self-driving car, a human support robot, etc.) faces many known and unknown task structures, our neuromodulatory goal-driven architecture should help it to choose tasks wisely regarding seen/unseen goals in a complex scenario.

Inspiration for Cognitive Neuroscience

Our experimental design could be replicated in biological studies with non-human primates or rodents to investigate relevant neuromodulatory signals in the brain. We predict that NE neurons would increase phasic activity after a goal switch. Corbetta et al. (2008) have shown that the locus coeruleus/norepinephrine system redirects attention from one object to another, and switches attention between networks. Attention is strongly modulated by acetylcholine through its projections to sensory cortex (Sarter et al., 2005). Cholinergic activation has been shown to increase goal-driven attention in V1 by increasing the firing rate of neurons coding the attended objects (Goard and Dan, 2009; Herrero et al., 2008). It would be of interest to test whether ACh activity to V1 becomes somewhat random after phasic NE responses and if ACh modulation varies depending on goal validity.

The robot experiments highlight a somewhat unexplored aspect of attention. In addition to feature or spatial attention, attention is deployed to intended actions (for a review, see Atkinson et al. (2018)). Recent results suggest that attention is required for both action planning and movement outcome monitoring (Mahon et al., 2018). In our robot experiments, an intended action led to attention to an object associated with the desired action. Such an attentional network could have benefits for human-robot interaction, especially when the intended actions can change due to context.

3.5 Conclusions

In this chapter, we introduced a model of ACh and NE neuromodulation to perform goal-driven perception. The proposed network architecture discovers goals using online learning, and highlights the stimulus features corresponding to the goal. Moreover, the proposed system rapidly adapts when goal contingencies change. This neurobiologically inspired model

can be applied to other problem domains and other top-down attentional networks.

Algorithm 1 ACh and NE Neuromodulation Process

Constant Input: $\beta = 0.7$, $num\_switches = 10$,
$K = 4$, $trial\_interval = 400$, $trial\_range = 30$,
$ne^{reset} = 0.25$, $ne^{min} = 0.25$, $ne^{max} = 1.0$,
$ch^{reset} = 1.0$, $ch^{min} = 0$, $ch^{max} = 10.0$,
$ne_{correct} = 0.70$, $ne_{wrong} = 1.10$,
$ch_{correct} = 1.40$, $ch_{wrong} = 0.90$
Other Input: all_test_pairs, $validity\_options$
Initialize ACh_i to ch^{reset} for $i = 1, 2, ..., K$.
Initialize NE to ne^{reset}.
Set $minLen$ to $(trial\_interval - trial\_range)$.
Set $maxLen$ to $(trial\_interval + trial\_range)$.
for $q = 1$ **to** $num\_switches$ **do**
    Randomly set *majorGoal* from 0, 1, ..., K-1.
    Set *minorGoal* from the same goal class.
    Randomly set *validity* from *validity_options*.
    Randomly set *trialLen* within $[minLen, maxLen]$.
    for $t = 1$ **to** *trialLen* **do**
        Pick a new *test_pair* from *all_pairs*.
        Randomly set r between [0, 1.0).
        if $r <$ *validity* **then**
            Set *trueGoal* to *majorGoal*.
        else
            Set *trueGoal* to *minorGoal*.
        end if
        Select *guessGoal* from Softmax (see Eqn. 3.4).
        Get *trueDigit* from *test_pair* with *trueGoal*.
        Apply *guessGoal* to the top layer.
        Obtain *map* via c-EB (see Eqn. 3.3).
        Get *predDigit* via fwd pass with *map*.
        Compare *predDigit* with *trueDigit*.
        Compare *trueGoal* with *guessGoal*.
        Update ACh and NE (see Eqn. 3.5 and Eqn. 3.6).
        Compute the reset threshold θ^{reset} (see Eqn. 3.7.)
        if $NE > \theta^{reset}$ **then**
            Reset ACh_i to ch^{reset} for $i = 1, 2, ..., K$.
            Reset NE to ne^{reset}.
        end if
    end for
end for

Chapter 4

Terrain Classification with a Reservoir-Based Network of Spiking Neurons

4.1 Introduction

Outdoor robots face many dynamic challenges that are uncommon in indoor scenarios. In particular, uneven terrain and a wide variety of surfaces found outdoors can lead to unpredictability. Different terrain types have an effect on robot movement and power usage. For any outdoor autonomous navigation system, the robot should have long-term path planning

strategies that consider trade-offs for traversing smooth surfaces, which may result in longer routes, versus direct routes that traverse over rough terrain, which may take more energy (Hwu et al., 2017a). Moreover, field robots need to operate over long periods of time far from power sources. In these cases, accurate terrain classification may be beneficial for navigation.

Neuromorphic architectures have potential for controlling outdoor robotics under tight power constraints. Unlike the traditional Von Neumann architecture, a neuromorphic architecture consumes less power due to massive parallelism and event-driven processing (Mead, 1990; Indiveri et al., 2011). Spiking neural networks (SNN) can take advantage of neuromorphic hardware, because each neuron computes its state independently, making the SNN parallel, and spikes are asynchronous events.

Navigation requires the effective use of a map. SLAM algorithms (Durrant-Whyte and Bailey, 2006) and GPS can provide solutions for navigation (Hwu et al., 2017b). However, these maps do not include terrain information, which is critical for planning trajectories. Therefore, accurate terrain classification can be an important addition to generate cost maps and help with real-time localization (Weszka et al., 1976; Manduchi et al., 2005; Lalonde et al., 2006; Mahadhir et al., 2014; Walas, 2015).

To address these challenges, this chapter introduces a reservoir-based spiking neural network (r-SNN) for terrain classification, which could be further integrated with other spiking navigation strategies to create a neuromorphic system for outdoor autonomous navigation.

4.2 Methods

4.2.1 Android Smartphone Solution

Android-Based Robotics Platform

Experiments were conducted using an Android-Based Robotics (ABR) Platform (see Figure 4.1). The GPS, accelerometer, gyroscope and visual information were directly obtained from an Android smartphone. A motor controller and IOIO-OTG microcontroller were mounted on the back of the platform. Communication between the phone and the robot platform was achieved through a Bluetooth connection with the IOIO-OTG. For robot specifications, see (Oros and Krichmar, 2013). The testing environment was a 19-acre botanical garden which contained different terrain types (i.e., grass, dirt, and road), different inclinations, and different obstacles (e.g., trees, benches, pedestrians, etc.).

Terrain Data Collection

The ABR robot was programmed to run at a constant speed over grass, dirt, and road terrains, labeled as 0, 1, 2 respectively, in the botanical garden. The data collection process

Figure 4.1: A six-wheel Android-based ground robot (ABR) used for terrain classification experiments.

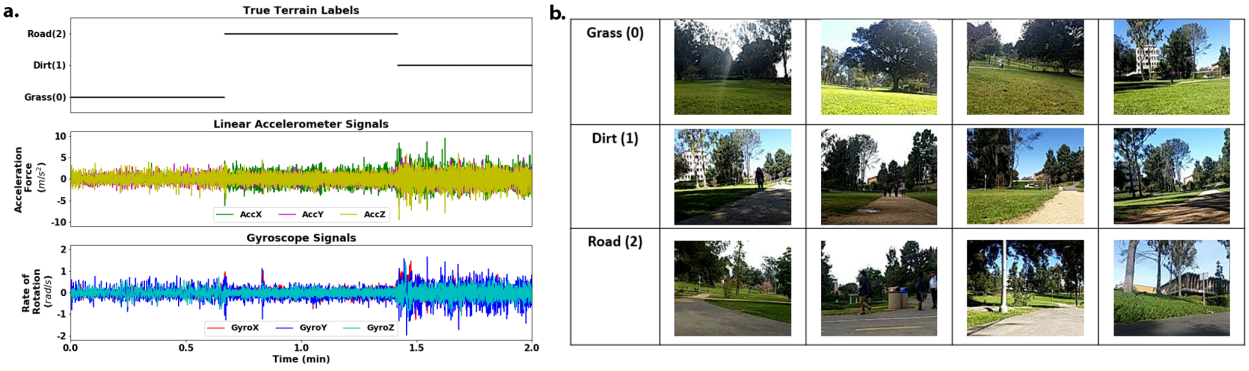


Figure 4.2: a) A sample trial with the original 3D linear accelerometer and gyroscope signals. b) Sample camera frames from the smartphone during data collection, with a resolution of 176×144 pixels. Each frame was cropped to keep only the bottom-center 5×5 pixels as terrain visual information.

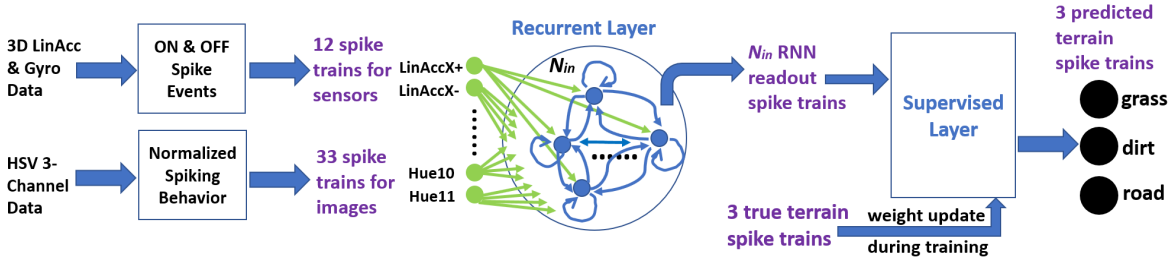


Figure 4.3: The terrain classification process with the r-SNN method.

was conducted under different lighting conditions during the daytime for 42 trials. Each trial lasted between 1 and 5 minutes. The 3-dimensional gyroscope and linear accelerometer data were collected at 100 Hz via the smartphone (see Figure 4.2a), and camera frames were captured at 20 Hz with a resolution of 176×144 pixels (see Figure 4.2b).

4.2.2 Reservoir-based Spiking Neural Network

Since our terrain classification algorithm might include both image and sensor input data, sequence memory and feature selection for both data types could be challenging for a feed-forward network. However, a recurrent neural network (RNN), where connections between internal neurons form a directed cycle, could use its internal state as the memory to process arbitrary sequences of inputs. RNNs have been used for a variety of applications, such as motion prediction, health monitoring, speech recognition, and time series forecasting (Kashyap et al., 2018; Das et al., 2018; Graves et al., 2013; Hewamalage et al., 2019). This recurrence can be tractably harnessed using a reservoir-based approach, such as Liquid State Machines (LSM) (Maass et al., 2002). In the LSM, the recurrent weights in the RNN are randomly generated and only the RNN readout is trained.

We developed a reservoir-based SNN (abbreviated to "r-SNN") method for terrain classification (see Figure 4.3 for the flow diagram). The readout from the recurrent layer (referred to here as RNN) was trained using a surrogate gradient approach that can learn using precise spike times in the LSM (Neftci et al., 2019). The ability of the r-SNN to classify terrains is compared with two conventional approaches, the Support Vector Machine (SVM) and the 3-layer (3L) logistic regression.

Spiking Neuron Model

The spiking neuron model for the recurrent and supervised layers consisted of leaky integrate-and-fire (LIF) neurons with current-based synaptic input. For each postsynaptic neuron i at each time step t, if it was not within the refractory period, the postsynaptic membrane potential (U_i) was updated via the differential equation

$$\frac{dU_i}{dt} = \frac{U^{rest} - U_i}{\tau^{mem}} + I_i^{syn}(t), \tag{4.1}$$

where U^{rest} was the resting membrane potential, τ^{mem} was the membrane time constant, and $I_i^{syn}(t)$ was the synaptic input current. $I_i^{syn}(t)$ jumped by summation of the weight w_{ij} upon spike arrival from each presynaptic neuron j (i.e., when $S_j(t) = 1$), with the equation shown below

$$\frac{d}{dt} I_i^{syn}(t) = -\frac{I_i^{syn}(t)}{\tau^{syn}} + \sum_{j \in pre} w_{ij} S_j(t). \tag{4.2}$$

When U_i reached the threshold θ^{mem} and the neuron i was not in the refractory period, a spike was triggered (i.e., $S_i(t) = 1$). The neuron then remained refractory for n^{ref} time steps.

Supervised Learning Rule

Inspired by SuperSpike (Zenke and Ganguli, 2018), during the supervised training process in which weight adaptation was requested (see Section 4.2.5), the synaptic weight w_{ij} was updated at each time step according to a nonlinear Hebbian rule with individual presynaptic traces ϵ_j,

$$\Delta w_{ij} = \eta \cdot [\epsilon_j \otimes (\hat{S}_i - \sigma(U_i))] \cdot \sigma(U_i) \cdot (1 - \sigma(U_i)), \tag{4.3}$$

where η was the learning rate, $\hat{S}_i$ was the target postsynaptic spiking behavior, and ϵ was a linear filter on the presynaptic spike activities. The portion $(\hat{S}_i - \sigma(U_i))$ represented the error signal. The presynaptic traces ϵ evolved according to

$$\frac{d\epsilon_j}{dt} = -\frac{\epsilon_j}{\tau^{syn}} + S_j(t). \tag{4.4}$$

With a small synaptic time constant τ^{syn}, this first-order filter was sufficient to evaluate the temporal convolution with the error signal in the expression of the presynaptic traces.

4.2.3 Spike Generation of Input Data for the Reservoir

To convert gyroscope and linear accelerometer sensor signals from the smartphone into spike trains, we used the same spike train encoding as in the the Dynamic Vision Sensor (DVS) (Yang et al., 2015). Six pairs of "plus" and "minus" spike trains for 3D signals of both sensors were converted into ON and OFF events, resulting in a total of 12 neurons for the sensor input data. For each axis of each sensor every time step, if the increase or decrease amount in the signal was above a threshold (i.e., 2 for the linear accelerometer and 0.5 for the gyroscope), an ON or OFF spike was generated, respectively.

Image data were converted from RGB (red, green, blue) to HSV (hue, saturation, value) and normalized between 0 and 1 for each channel. Each frame was cropped to keep only the bottom-center 5×5 pixels, which was enough to show the current terrain visual information without interference from distractors in the scene (e.g., other terrain types, trees, benches, pedestrians, or buildings). Each HSV channel was averaged across all 25 pixels in the image. There were 11 neurons for each HSV channel (i.e., a total of 33 neurons). Each neuron's activity was based on a Gaussian tuning curve. The means of the Gaussians were spread evenly 0 to 1 with $\sigma = 0.5$. The maximum activity for each tuning curve was $\alpha = 1/(\sigma\sqrt{2\pi})$. If activity was above 0.4α, the neuron spiked.

The frequencies of the sensor signals and camera frames were different (i.e., 100 Hz and 20 Hz respectively). Therefore when both sensor signals and image frame were fed into the recurrent layer, the same image frame would be repeated for each time step until the next frame was collected.

4.2.4 Recurrent Layer for Terrain Feature Extraction

The gyroscope signals, the linear accelerometer signals, and the cropped screenshots were encoded directly into spikes, as described above. These input spikes were fed into the recurrent layer. The sensor and image neurons were fully connected with the recurrent neurons. The recurrent neurons were also fully connected with one another. For this recurrent layer, the synaptic input current was a summation of both input weights and recurrent weights when spikes were received. The readout spikes from the recurrent layer were further fed into the supervised layer for terrain classification (see Section 4.2.5).

There were $N_{in} = 70$ recurrent neurons, which received N_{ext} input spike trains from the sensors and/or images (i.e., $N_{ext} = 12$, 33, 45 respectively). Both input and recurrent weights were randomly drawn from a Gaussian distribution with zero mean. The standard deviation was $0.5/N_{ext}$ for input weights or $0.05/N_{in}$ for the recurrent weights, which was small enough to prevent bias on certain connections while assuring randomness. Therefore recurrent neurons were both excitatory and inhibitory. In Equation 4.1, U^{rest} and the initial U_i were both 0, whereas τ^{mem} was tuned to 66.7. In Equation 4.2, the initial $I_i^{syn}(t)$ was 0, whereas τ^{syn} was tuned to 1. Furthermore, θ^{mem} was $(1 - \gamma)$, with γ as the threshold Gaussian noise with mean at 0 and a standard deviation of 0.1. n^{ref} was zero, meaning there was no refractory period.

4.2.5 Supervised Layer for Terrain Classification

The supervised layer took as input the 70 readout spike trains from the recurrent neurons and generated spike activities for the three terrain prediction output neurons that represented grass, dirt, and road. The output weights were updated every time step (see Equation 4.3) for 100 training epochs and remained constant for testing. During the training process, the target postsynaptic spiking behavior was obtained from three spikes trains that represented actual terrain information at each time step. For the supervised layer, the synaptic input current evolved with summation of output weights upon spike arrival. The postsynaptic neurons were the three terrain prediction neurons. A terrain class was predicted by the output neuron with the highest activity at that time step.

Before the first training epoch, the output weights were all initialized as $0.001/N_{in} = 1\text{e}{-5}$ so that all the readout spikes could excite the three terrain output neurons. In Equation 4.1, U^{rest} was 0 and τ^{mem} was tuned to 100, but U_i was initialized as -0.5 for each epoch. In Equations 4.2 and 4.4, τ^{syn} was tuned to 10, whereas $I_i^{syn}(t = 0)$ was initialized as zero at the beginning of each training epoch and of testing. θ^{mem} and n^{ref} were the same as in Section 4.2.4. In Equation 4.3, η was tuned to $9\text{e}{-9}$. The sigmoid function was tuned to $\sigma(x) = 1/(1 + \exp[-3.44 \cdot (x - 0.975)])$.

4.3 Results

4.3.1 Terrain Prediction Results for the r-SNN

The r-SNN method achieved over 90% of testing accuracy in predicting different terrain types, with either linear accelerometer and gyroscope sensors, or image inputs, or both sensor and image inputs (see Table 4.1). Figures 4.4 and 4.5 show results using both images

and sensors for an 8-minute testing period. From the 70 recurrent neurons (see Figure 4.4), the supervised layer generated output spikes for terrain classification. After 100 training epochs, the test prediction consistently matched the true terrain with little delay or noise (see Figure 4.5).

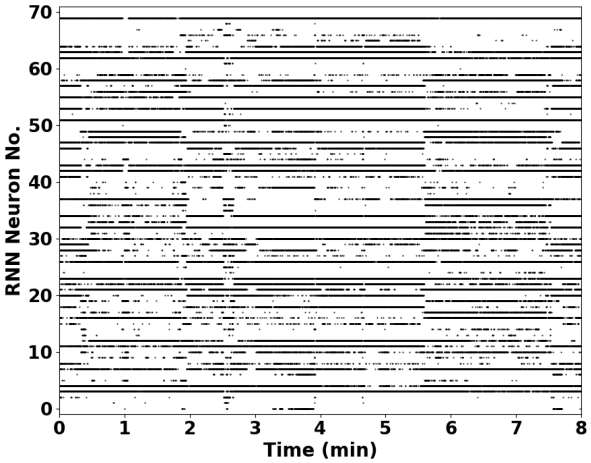


Figure 4.4: Readout spikes from all the recurrent neurons using both image and sensor (the linear accelerometer and gyroscope) inputs. The horizontal axis labels partial testing period of 8 min, with sensor signals collected at 100 Hz and camera frames collected at 20 Hz. These 70 readout spike trains were further fed into the supervised testing part for terrain classification.

4.3.2 Optimal Settings for Two Conventional Approaches

For the SVM model and the 3L logistic regression model, we split the original sensor and image signals into data chunks with a time window of 500 milliseconds. The optimal performance on the SVM model was achieved by using the SVC package in the Scikit-learn library with the RBF kernel (Pedregosa et al., 2011). The SVM applied the nine features: (1) number of sign changes, (2) number of traverses over mean, (3) standard deviation, (4) autocorrelation at lag k=1, (5) maximum, (6) minimum, (7) Euclidean norm, (8) mean, and (9) median. Its best test accuracy was achieved after 430 training epochs.

The optimal performance on the 3L logistic regression model required the following five

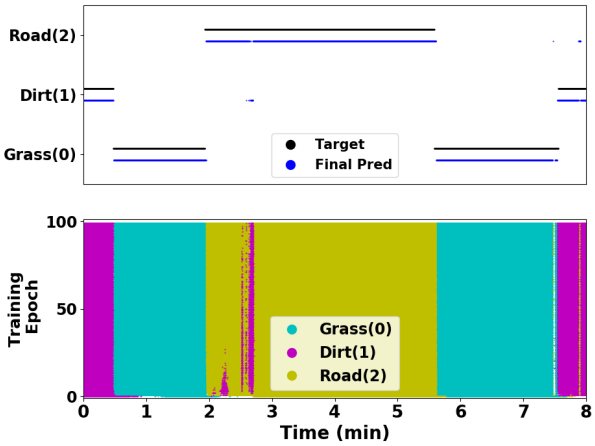


Figure 4.5: Supervised layer output for both images and sensors (the linear accelerometer and gyroscope) for an 8-minute testing period. The upper subplot shows true terrain types and final test predictions using adapted output weights after 100 training epochs. The lower subplot shows the test prediction spikes using adapted output weights after each training epoch.

features: (1) 20-percentile, (2) 50-percentile, (3) 80-percentile, (4) mean, (5) standard deviation. Its best test accuracy was achieved after 520 training epochs by using the mean squared error as the loss function, or after 820 training epochs with the cross-entropy loss function.

4.3.3 Model Performance Comparison

For comparison among three approaches on terrain classification, we applied the standard 80/20 rule for training and testing. The input data were generated from linear accelerometer and gyroscope sensor signals and/or cropped screenshots.

Table 4.1 shows the test error rates among three approaches under three input conditions. Using both image and sensor data instead of using one of them improved the accuracy and robustness of each model. The r-SNN method was more accurate than SVM and 3L logistic regression. The r-SNN may be the most efficient considering its usage of only 70 recurrent

neurons, adaptation of only the output weights, and no need of splitting data into time chunks.

The r-SNN is compatible with low-power neuromorphic architectures, whose energy cost is often dominated by synaptic operations (SynOps), akin to multiply accumulate operations (MACs) in digital computers for artificial networks (Merolla et al., 2014; Neftci et al., 2016). For our entire training and test process with terrain classified every 500 milliseconds, the r-SNN would require roughly 10^9 SynOps on a neuromorphic hardware, which is equal to or smaller than the operations taken by the 3L logistic regression and the SVM (i.e., roughly $10^9 \sim 10^{10}$ MACs for each) on a standard computer. Based on the fact that a SynOp consumes many fold less energy than a MAC (Merolla et al., 2014; Davies et al., 2018; Neftci et al., 2017), the r-SNN is a promising approach to reduce power consumption.

Table 4.1: Test error rates on three models for terrain classification.

	r-SNN	**SVM**	**3L Logistic Regression**	
			mse	**xent**
Images only	**5.2%**	13.9%	11.5%	16.2%
Sensors only	**8.1%**	14.5%	13.7%	59.6%
Images + Sensors	**3.5%**	8.8%	10.2%	34.3%

4.4 Conclusion

Unlike feed-forward networks, the recurrent layer processes both the sensor and image input data to extract abstract terrain features at each time step, with no need of remembering data chunks within a time window or carefully selecting feature components. The reservoir computing paradigm lowers the computational cost during supervised training, because only the output weights are plastic (Maass et al., 2002). Moreover, having spiking neurons in the reservoir allows the model to be event-driven and highly parallel. Further performance

gains can be achieved by implementing the present algorithm on neuromorphic hardware that utilizes spike-based strategies.

The r-SNN has several advantages for classification tasks such as discriminating terrains. First, it had the highest test prediction accuracy compared to the SVM and 3L logistic regression, regardless of whether images or sensors were the input (see Table 4.1). Secondly, it had the lowest computational cost due to a small reservoir of spiking neurons and adaptation of only the output weights. Third, compared to the difficulty in selecting the terrain features and time window length for the two conventional approaches, the r-SNN reservoir can easily integrate the image and/or sensor data and generate an abstract representation of terrain features.

The trained r-SNN model is compatible with a ground robot for real-time terrain classification. The r-SNN can be used to augment a SLAM or GPS map with metadata pertaining to the cost of traversal. For example, the r-SNN can supplement a road following algorithm to signal when the robot veers off the road. The different terrains can be used as a cost function, based on terrain smoothness for path planning (Hwu et al., 2017b). Finally, the r-SNN presented here can be used in to develop a complete neuromorphic robot navigation system capable of operating over long durations with minimal power consumption (Hwu et al., 2017a; Kreiser et al., 2018).

Chapter 5

Future Directions

5.1 Better Understanding of Animal Behavior and Brain Activities

The evolved RNN in our cognitive map project (Zou et al., 2021) has similar characteristics to the hippocampus. We have demonstrated that a robot controlled by an evolved RNN could solve a spatial and working memory task where the robot needed to navigate a maze and remember not to repeat paths it had taken previously. The RNN populational activity carried spatial information sufficient to localize the robot (i.e., spatial memory) as well as a prospective code of where the robot intent to go next (i.e., working memory and planning). Similar to CA1, the RNN received speed, direction, and visual information as input, and combined these types of sensory information to construct a journey-dependent place code (Taube et al., 1990; Sargolini et al., 2006; Kropff et al., 2015; O'Keefe, 1976; Hafting et al., 2005; Sun et al., 2019; Potvin et al., 2007; Frost et al., 2020). Moreover, the robotic behavior was dependent on RNN dynamics rather than a sensor-to-motor mapping (Zou et al., 2021). We are also exploring "latent learning", one of Tolman's cognitive map ideas (Tolman,

1948), to understand more about the spatial and working memory encoding in this biological plausible network. Current results suggest that the agent, which did not display its learning effect during maze exploration until there were reward stimuli, actually performs better than the other agent which received reward stimuli since the beginning.

For the next step, we would like to utilize the major benefits of our existing network but modify the paradigm to co-evolve multiple brain regions (e.g., CA1, SUB, and PFC or RSC) so that we can solve a more comprehensive navigational task. Although this plan would operate in a much larger scale than our existing system, it is decomposable. We can start with only co-evolving the connections between CA1 and SUB while freezing the PFC or RSC component, and then freeze another and co-evolve the rest. We might use the regular RNN as we did before or try with the LSTM to better link experiences over a long duration by memorizing the most important information. By doing so, we expect to evolve not only place coding and head direction but also more abstract representation of location, direction, distance, and speed in grid cell activities. These cell types may occur in one or multiple evolved regions. By comparing their activities with biological data and theories, we might be able to replicate or even further explain the neural activities in multiple regions of an animal's brain.

Furthermore, regarding the neuromodulatory systems in the brain, we have explored a few typical ones but still can find a large space to improve. We utilized the cholinergic (ACh) and noradrenergic (NE) systems for expected and unexpected uncertainties in our goal-driven perception project (Zou et al., 2020b). We also used the serotonin (5-HT) system for patience control in an outdoor waypoint navigation task (Xing et al., 2020). However, our existing neuromodulatory settings are not flexible enough to face dynamic situations. In the first problem, the number of ACh neurons was preset for a fixed number of goals. We should modify the system to judge if a goal is familiar or completely novel and assign a new ACh neuron every time an unknown goal appears. In the second problem, the 5-HT

activity level could be only chosen between the two fixed modes for a more assertive or more patient behavior. We should regulate the 5-HT activity level in a continuous manner to build a more intelligent navigational control scheme. The models of neuromodulation could be further refined in the future by either collaborating closely with a neurobiological research group that looks into detection and analysis of neuromodulatory signals in the rat's or primate's brain.

5.2 Utilization of Detailed Simulators

Many experiments are computationally expensive or relatively more difficult to directly apply on physical robots. In such cases, some robotic simulators can provide efficient system control and signal processing without loss of much accuracy or realism in the simulated robot design. Because of the accurate representation of many popular robot models in the Webots simulator, we could always run much faster than real-time (e.g., $\times 30$ faster on average on our Alienware desktop with one Nvidia GeForce GTX 1080 Ti) to evolve for enough generations before applying a well-performed genotype to the RNN controller for real-world robotic navigation tasks. The power of using a detailed simulator, such as Webots, is that the trained controller should transfer to the real robot with minimal adjustments. By having experience with an intensive cognitive map project in Webots, it is also easier for us to experiment with unfamiliar robot models in the same or altered virtual environment. For example, before using the Toyota Human Support Robot (HSR) for path planning in our actual building, we can first try different navigational strategies with the PR2 robot model in a virtual indoor setting.

Regarding self-driving in the CARLA simulator, we are working in an RL paradigm to test our neuromodulated attention and task-driven perception with perturbation-based saliency maps. There are currently two tasks, aggressive driving and careful driving, but we can

experiment on others (e.g., point-to-point navigation, reactive strategies for construction zones or different weather conditions, etc.) that utilize more digital assets in this simulator.

5.3 Robotic Assistance for Human Daily Life

In the real world, robotic assistance is very important for human daily life. Some people such as the elderly, bed-bound patients, and those with visual impairments can often experience inconvenience or even incapabilities of moving to a desired location, grabbing certain stuff, or achieving some other daily tasks. Some children with heart conditions or immune deficiency have the potential to learn, but cannot be near other children, and thus are homebound from school. Even for normal individuals, we may be busy with the current working or study items, but also want to complete some side tasks at the same time to improve efficiency. In all these cases, a human support robot that is fully autonomous or with minimal requirement of manual control can represent the customer to resolve those daily requests.

Our neuromodulated goal-driven perception model can be viewed as a possible approach when designing such robotic system. As shown in our action-based attention setting, the Toyota HSR was able to guess the action (i.e., "eat", "work-on-computer", "read", "say-hi") from the neuromodulatory neural network and direct attention to the object in the scene that could achieve that action (Zou et al., 2020b). In the future, we can further integrate this model with our navigational strategies (Zou et al., 2020a, 2021; Hwu et al., 2017a, 2020) to achieve more complicated robotic tasks. For example, the robot can not only pay attention to neuromodulated goal-related objects but also efficiently navigate through a complex structure to categorize different room schema. Its working memory can help it to remember what has done and what to do next.

5.4 Autonomous Navigation and Path Planning

Autonomous robot navigation requires integration between long-term path planning strategies and reactive strategies (e.g., road following, terrain classification, and obstacle avoidance). We have implemented a few of them via our Android Based Robot (ABR) (Zou et al., 2020a; Hwu et al., 2017b,a; Xing et al., 2020). Improvements can be done in the following aspects: (1) Our reservoir-based SNN for terrain classification (Zou et al., 2020a) can serve for real-time update of the cost map for path planning in a dynamic or unfamiliar outdoor environment. (2) Our neuromodulated attention model (Zou et al., 2020b) can be adapted to focus on only important aspects in the front camera view regarding the current goal. (3) We can evolve an RNN in a similar way to what we did in the cognitive map project (Zou et al., 2021) to encode spatial memory (i.e., location, direction, and distance information) and working memory (i.e., where has been visited and where to go next). (4) We may also want to further explore the continuous regulation of multiple neuromodulators (e.g., acetylcholine, norepinephrine, dopamine, serotonin, etc.) for assessment of danger, reward, novelty and regulation of patience, assertiveness, surprise. (5) Because the IOIO board used to control our ABR is no longer supportive, we should plan future physical robot experiments with ROS-based platforms (e.g., Jackal UGV, Toyota HSR, etc.). (6) For energy reduction, we can apply our spike-based algorithms on a digital neuromorphic chip such as Intel Loihi (Davies et al., 2018) and integrate it with a robot platform.

Furthermore, studies have shown that different people have varied navigational preferences to reach the target destination from the same origin (Weisberg and Newcombe, 2016, 2018). We can develop a multi-agent system that utilizes the benefits of such populational variability. In the exploration (early) stage of a task (e.g., search-and-rescue), this strategy can improve information gathering. In the exploitation (late) stage, these agents can efficiently solve the task in a self-assembly manner with complex cooperation and functional specialization (Dorigo et al., 2013; Ducatelle et al., 2011).

Chapter 6

Conclusions

All my doctoral research projects involve neurorobotic investigation of various artificial networks inspired from neurobiology. In the opposite direction, they also lead to a qualitative and quantitative explanation of real brain activities. For the cognitive map project, we demonstrated that an evolved RNN, which linked the sensor values of a simulated robot to its motor wheel output, could replicate the rat's neural activities in a maze-navigation task. The RNN population carried spatial information sufficient to localize robot in the triple T-maze (i.e., spatial memory) and prospective predictive information of which path robot intended on taking (i.e., working memory). Current results also suggest that latent learning (i.e., learning without display until there is a motivation) does occur in our evolved RNN. For the neuromodulated goal-driven perception project, our model was inspired by the effects of cholinergic and noradrenergic neuromodulatory systems on attention and tracking uncertainties. The Toyota Human Support Robot was able to guess the action from the neuromodulatory heads and direct attention to the object in the scene that could achieve that action. For the terrain classification project, our reservoir-based SNN received spike inputs from linear accelerometer, gyroscope, and camera signals collected by the Android Based Robot. Only its output weights were adapted to generate accurate prediction from three

terrain types in a botanical garden. It is also compatible with neuromorphic hardware which can significantly reduce energy cost with event-driven, parallel computing. What have been discussed in my dissertation are all adaptable to the UCI CARL lab's future projects and my future career. Furthermore, they can all be improved and contribute to the autonomous lifelong learning AI, in a way more similar to biological intelligence, to help with daily-life assistance, navigation, planning, and other practical applications.

Bibliography

J. Achiam and S. Sastry. Surprise-based intrinsic motivation for deep reinforcement learning. *arXiv preprint arXiv:1703.01732*, 2017.

K. Akinci and A. Philippides. Evolving recurrent neural network controllers by incremental fitness shaping. In *ALIFE 2019: Proceedings of the Artificial Life Conference 2019*, pages 416–423, Newcastle, United Kingdom, July 2019. MIT Press. doi: 10.1162/isal_a_00196.

F. Akopyan, J. Sawada, A. Cassidy, R. Alvarez-Icaza, J. Arthur, P. Merolla, N. Imam, Y. Nakamura, P. Datta, G.-J. Nam, et al. Truenorth: Design and tool flow of a 65 mw 1 million neuron programmable neurosynaptic chip. *IEEE transactions on computer-aided design of integrated circuits and systems*, 34(10):1537–1557, 2015.

P. Andersen, R. Morris, D. Amaral, J. O'Keefe, and T. Bliss. *The hippocampus book*. Oxford university press, 2007.

M. A. Atkinson, A. A. Simpson, and G. G. Cole. Visual attention and action: How cueing, direct mapping, and social interactions drive orienting. *Psychonomic bulletin & review*, 25(5):1585–1605, 2018.

M. C. Avery, N. Dutt, and J. L. Krichmar. Mechanisms underlying the basal forebrain enhancement of top-down and bottom-up attention. *Eur J Neurosci*, 39(5):852–865, 2014. ISSN 1460-9568 (Electronic) 0953-816X (Linking). doi: 10.1111/ejn.12433.

A. D. Baddeley and G. J. Hitch. Developments in the concept of working memory. *Neuropsychology*, 8(4):485—493, 1994.

T. Bailey and H. Durrant-Whyte. Simultaneous localization and mapping (slam): Part ii. *IEEE robotics & automation magazine*, 13(3):108–117, 2006.

F. Baluch and L. Itti. Mechanisms of top-down attention. *Trends in neurosciences*, 34(4): 210–224, 2011.

A. Banino, C. Barry, B. Uria, C. Blundell, T. Lillicrap, P. Mirowski, A. Pritzel, M. J. Chadwick, T. Degris, J. Modayil, et al. Vector-based navigation using grid-like representations in artificial agents. *Nature*, 557(7705):429–433, 2018.

C. Barry, R. Hayman, N. Burgess, and K. J. Jeffery. Experience-dependent rescaling of entorhinal grids. *Nature neuroscience*, 10(6):682, 2007.

M. G. Baxter and A. A. Chiba. Cognitive functions of the basal forebrain. *Current opinion in neurobiology*, 9(2):178–183, 1999.

J. C. Bongard. Evolutionary robotics. *Communications of the ACM*, 56(8):74–83, Aug 2013. doi: 10.1145/2493883.

S. Bouret and S. J. Sara. Network reset: a simplified overarching theory of locus coeruleus noradrenaline function. *Trends in neurosciences*, 28(11):574–582, 2005.

G. Brockman, V. Cheung, L. Pettersson, J. Schneider, J. Schulman, J. Tang, and W. Zaremba. Openai gym, 2016.

Y. Burda, H. Edwards, D. Pathak, A. Storkey, T. Darrell, and A. A. Efros. Large-scale study of curiosity-driven learning. *arXiv preprint arXiv:1808.04355*, 2018.

C. Cao, X. Liu, Y. Yang, Y. Yu, J. Wang, Z. Wang, Y. Huang, L. Wang, C. Huang, W. Xu, et al. Look and think twice: Capturing top-down visual attention with feedback convolutional neural networks. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 2956–2964, 2015.

J. Charvet, L. Shu, S. Laurent-Charvet, B. Wang, M. Faure, D. Cluzel, Y. Chen, and K. De Jong. Palaeozoic tectonic evolution of the tianshan belt, nw china. *Science China Earth Sciences*, 54(2):166–184, 2011.

K. Chen, T. Hwu, H. J. Kashyap, J. L. Krichmar, K. Stewart, J. Xing, and X. Zou. Neurorobots as a means toward neuroethology and explainable ai. *Frontiers in Neurorobotics*, 14, 2020.

L. L. Chen, L.-H. Lin, E. J. Green, C. A. Barnes, and B. L. McNaughton. Head-direction cells in the rat posterior cortex. *Experimental brain research*, 101(1):8–23, 1994.

K. Cho, A. Courville, and Y. Bengio. Describing multimedia content using attention-based encoder-decoder networks. *IEEE Transactions on Multimedia*, 17(11):1875–1886, 2015.

J. Clune, B. E. Beckmann, C. Ofria, and R. T. Pennock. Evolving coordinated quadruped gaits with the hyperneat generative encoding. In *2009 IEEE Congress on Evolutionary Computation*, pages 2764–2771, Trondheim, Norway, 2009. IEEE. doi: 10.1109/CEC.2009.4983289.

M. Corbetta, G. Patel, and G. L. Shulman. The reorienting system of the human brain: from environment to theory of mind. *Neuron*, 58(3):306–324, 2008.

A. B. Craig, M. E. Phillips, A. Zaldivar, R. Bhattacharyya, and J. L. Krichmar. Investigation of biases and compensatory strategies using a probabilistic variant of the wisconsin card sorting test. *Frontiers in Psychology*, 7(17), 2016. ISSN 1664-1078. doi: 10.3389/fpsyg.2016.00017.

A. Das, P. Pradhapan, W. Groenendaal, P. Adiraju, R. T. Rajan, F. Catthoor, S. Schaafsma, J. L. Krichmar, N. Dutt, and C. Van Hoof. Unsupervised heart-rate estimation in wearables with liquid states and a probabilistic readout. *Neural Networks*, 99:134–147, 2018.

M. Davies, N. Srinivasa, T.-H. Lin, G. Chinya, Y. Cao, S. H. Choday, G. Dimou, P. Joshi, N. Imam, S. Jain, et al. Loihi: A neuromorphic manycore processor with on-chip learning. *IEEE Micro*, 38(1):82–99, 2018.

P. Dayan and L. F. Abbott. *Theoretical neuroscience: computational and mathematical modeling of neural systems*. Computational Neuroscience Series, 2001.

P. Dayan and A. Yu. Phasic norepinephrine: A neural interrupt signal for unexpected events. *Network: Computation in Neural Systems*, 17:335–350, 2006. ISSN 0954-898X.

A. Diamond. Executive functions. *Annual Review of Psychology*, 64(1):135–168, 2013. doi: 10.1146/annurev-psych-113011-143750.

M. Dorigo, D. Floreano, L. M. Gambardella, F. Mondada, S. Nolfi, T. Baaboura, M. Birattari, M. Bonani, M. Brambilla, A. Brutschy, et al. Swarmanoid: a novel concept for the study of heterogeneous robotic swarms. *IEEE Robotics & Automation Magazine*, 20(4): 60–71, 2013.

A. Dosovitskiy, G. Ros, F. Codevilla, A. Lopez, and V. Koltun. Carla: An open urban driving simulator. In *Conference on robot learning*, pages 1–16. PMLR, 2017.

K. L. Downing. *Intelligence emerging: adaptivity and search in evolving neural systems*. MIT Press, 2015.

K. Doya. Metalearning and neuromodulation. *Neural Networks*, 15(4):495–506, 2002. ISSN 0893-6080. doi: https://doi.org/10.1016/S0893-6080(02)00044-8.

F. Ducatelle, G. A. Di Caro, C. Pinciroli, and L. M. Gambardella. Self-organized cooperation between robotic swarms. *Swarm Intelligence*, 5(2):73–96, 2011.

H. Durrant-Whyte and T. Bailey. Simultaneous localization and mapping: part i. *IEEE robotics & automation magazine*, 13(2):99–110, 2006.

D. A. Eldreth, M. D. Patterson, A. J. Porcelli, B. B. Biswal, D. Rebbechi, and B. Rypma. Evidence for multiple manipulation processes in prefrontal cortex. *Brain Research*, 1123 (1):145–156, 2006.

B. R. Fajen and W. H. Warren. Behavioral dynamics of steering, obstable avoidance, and route selection. *Journal of Experimental Psychology: Human Perception and Performance*, 29(2):343–362, 2003. doi: 10.1037/0096-1523.29.2.343.

J. Ferbinteanu and M. L. Shapiro. Prospective and retrospective memory coding in the hippocampus. *Neuron*, 40(6):1227–1239, Dec 2003. ISSN 0896-6273. doi: 10.1016/S0896-6273(03)00752-9.

D. Floreano and L. Keller. Evolution of adaptive behaviour in robots by means of darwinian selection. *PLOS Biology*, 8(1):1–8, 2010.

D. B. Fogel. The advantages of evolutionary computation. In *Bcec*, pages 1–11, 1997.

G. B. Fogel and D. W. Corne. *Evolutionary computation in bioinformatics*. Elsevier, 2002.

L. J. Fogel, A. J. Owens, and M. J. Walsh. Artificial intelligence through simulated evolution. 1966.

B. E. Frost, S. K. Martin, M. Cafalchio, M. N. Islam, J. P. Aggleton, and S. M. O'Mara. Anterior thalamic function is required for spatial coding in the subiculum and is necessary for spatial memory, 2020.

J. Gayon. *Darwinism's struggle for survival: heredity and the hypothesis of natural selection*. Cambridge University Press, 1998.

W. Gerstner and W. M. Kistler. *Spiking neuron models: Single neurons, populations, plasticity*. Cambridge university press, 2002.

L. M. Giocomo, T. Stensola, T. Bonnevie, T. Van Cauter, M.-B. Moser, and E. I. Moser. Topography of head direction cells in medial entorhinal cortex. *Current Biology*, 24(3): 252–262, 2014.

M. Goard and Y. Dan. Basal forebrain activation enhances cortical coding of natural scenes. *Nature neuroscience*, 12(11):1444, 2009.

A. Graves, A.-r. Mohamed, and G. Hinton. Speech recognition with deep recurrent neural networks. In *2013 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 6645–6649, Vancouver, BC, Canada, 2013. IEEE. doi: 10.1109/ICASSP.2013.6638947.

S. Greengard. Neuromorphic chips take shape. *Communications of the ACM*, 63(8):9–11, 2020.

S. L. Grella, J. M. Neil, H. T. Edison, V. D. Strong, I. V. Odintsova, S. G. Walling, G. M. Martin, D. F. Marrone, and C. W. Harley. Locus coeruleus phasic, but not tonic, activation initiates global remapping in a familiar environment. *Journal of Neuroscience*, 39(3):445–455, 2019.

S. Greydanus. Excitationbp: visualizing how deep networks make decisions. `https://github.com/greydanus/excitationbp`, 2018.

T. Hafting, M. Fyhn, S. Molden, M.-B. Moser, and E. I. Moser. Microstructure of a spatial map in the entorhinal cortex. *Nature*, 436(7052):801–806, 2005. doi: 10.1038/nature03721.

N. Hansen. The cma evolution strategy: a comparing review. *Towards a new evolutionary computation*, pages 75–102, 2006.

J. L. Herrero, M. Roberts, L. S. Delicato, M. A. Gieselmann, P. Dayan, and A. Thiele. Acetylcholine contributes through muscarinic receptors to attentional modulation in v1. *Nature*, 454(7208):1110, 2008.

H. Hewamalage, C. Bergmeir, and K. Bandara. Recurrent neural networks for time series forecasting: Current status and future directions. *arXiv preprint arXiv:1909.00590*, 2019.

S. Hochreiter and J. Schmidhuber. Long short-term memory. *Neural computation*, 9(8): 1735–1780, 1997.

A. L. Hodgkin and A. F. Huxley. The components of membrane conductance in the giant axon of loligo. *The Journal of physiology*, 116(4):473–496, 1952a.

A. L. Hodgkin and A. F. Huxley. Currents carried by sodium and potassium ions through the membrane of the giant axon of loligo. *The Journal of physiology*, 116(4):449–472, 1952b.

A. L. Hodgkin and A. F. Huxley. A quantitative description of membrane current and its application to conduction and excitation in nerve. *The Journal of physiology*, 117(4): 500–544, 1952c.

A. L. Hodgkin, A. F. Huxley, and B. Katz. Measurement of current-voltage relations in the membrane of the giant axon of loligo. *The Journal of physiology*, 116(4):424–448, 1952.

J. H. Holland. Genetic algorithms. *Scientific american*, 267(1):66–73, 1992.

T. Hwu, J. Krichmar, and X. Zou. A complete neuromorphic solution to outdoor navigation and path planning. In *2017 IEEE International Symposium on Circuits and Systems (ISCAS)*, pages 1–4. IEEE, 2017a.

T. Hwu, A. Y. Wang, N. Oros, and J. L. Krichmar. Adaptive robot path planning using a spiking neuron algorithm with axonal delays. *IEEE Transactions on Cognitive and Developmental Systems*, 10(2):126–137, 2017b.

T. Hwu, H. J. Kashyap, and J. L. Krichmar. A neurobiological schema model for contextual awareness in robotics. In *2020 International Joint Conference on Neural Networks (IJCNN)*, pages 1–8. IEEE, 2020.

A. J. Ijspeert. Biorobotics: Using robots to emulate and investigate agile locomotion. *science*, 346(6206):196–203, 2014.

G. Indiveri, B. Linares-Barranco, T. J. Hamilton, A. Van Schaik, R. Etienne-Cummings, T. Delbruck, S.-C. Liu, P. Dudek, P. Häfliger, S. Renaud, et al. Neuromorphic silicon neuron circuits. *Frontiers in neuroscience*, 5:73, 2011.

L. Itti and C. Koch. A saliency-based search mechanism for overt and covert shifts of visual attention. *Vision research*, 40(10-12):1489–1506, 2000.

E. M. Izhikevich. Simple model of spiking neurons. *IEEE Transactions on neural networks*, 14(6):1569–1572, 2003.

E. M. Izhikevich. Which model to use for cortical spiking neurons? *IEEE transactions on neural networks*, 15(5):1063–1070, 2004.

H. Jaeger. Echo state network. *scholarpedia*, 2(9):2330, 2007.

Y. Jia, E. Shelhamer, J. Donahue, S. Karayev, J. Long, R. Girshick, S. Guadarrama, and T. Darrell. Caffe: Convolutional architecture for fast feature embedding. In *Proceedings of the 22nd ACM international conference on Multimedia*, pages 675–678. ACM, 2014.

H. J. Kashyap, G. Detorakis, N. Dutt, J. L. Krichmar, and E. Neftci. A recurrent neural network based model of predictive smooth pursuit eye movement in primates. In *2018 International Joint Conference on Neural Networks (IJCNN)*, pages 1–8, Rio de janeiro, Brazil, 2018. IEEE.

D. P. Kingma and J. Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

J. R. Koza. *Genetic programming II: automatic discovery of reusable programs*. MIT press, 1994.

J. R. Koza and J. R. Koza. *Genetic programming: on the programming of computers by means of natural selection*, volume 1. MIT press, 1992.

J. R. Koza, M. A. Keane, M. J. Streeter, W. Mydlowec, J. Yu, and G. Lanza. *Genetic programming IV: Routine human-competitive machine intelligence*, volume 5. Springer Science & Business Media, 2006.

R. Kreiser, A. Renner, Y. Sandamirskaya, and P. Pienroj. Pose estimation and map formation with spiking neural networks: Towards neuromorphic slam. In *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 2159–2166. IEEE, 2018.

J. L. Krichmar. The neuromodulatory system: A framework for survival and adaptive behavior in a challenging world. *Adaptive Behavior*, 16(6):385–399, 2008. doi: 10.1177/1059712308095775.

J. L. Krichmar. Neurorobotics-a thriving community and a promising pathway toward intelligent cognitive robots. *Frontiers in Neurorobotics*, 12, 2018. ISSN 1662-5218. doi: ARTN4210.3389/fnbot.2018.00042.

J. L. Krichmar, T. Hwu, X. Zou, and T. Hylton. Advantage of prediction and mental imagery for goal-directed behaviour in agents and robots. *Cognitive Computation and Systems*, 1 (1):12–19, 2019.

E. Kropff, J. E. Carmichael, M.-B. Moser, and E. I. Moser. Speed cells in the medial entorhinal cortex. *Nature*, 523(7561):419–424, 2015.

J.-F. Lalonde, N. Vandapel, D. F. Huber, and M. Hebert. Natural terrain classification using three-dimensional ladar data for ground robot mobility. *Journal of field robotics*, 23(10): 839–861, 2006.

Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

X. Li and R. Miikkulainen. Opponent modeling and exploitation in poker using evolved recurrent neural networks. In *GECCO '18 Companion*, pages 189–196, Kyoto, Japan, 2018. ACM. doi: 10.1145/3205455.3205589.

T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick. Microsoft coco: Common objects in context. In *European conference on computer vision*, pages 740–755. Springer, 2014.

R. Lock, S. Burgess, and R. Vaidyanathan. Multi-modal locomotion: from animal to application. *Bioinspiration & biomimetics*, 9(1):011001, 2013.

W. Maass, T. Natschläger, and H. Markram. Real-time computing without stable states: A new framework for neural computation based on perturbations. *Neural computation*, 14 (11):2531–2560, 2002.

K. A. Mahadhir, S. C. Tan, C. Y. Low, R. Dumitrescu, A. T. M. Amin, and A. Jaffar. Terrain classification for track-driven agricultural robots. *Procedia Technology*, 15:775–782, 2014.

A. Mahon, S. Bendžiūtė, C. Hesse, and A. R. Hunt. Shared attention for action selection and action monitoring in goal-directed reaching. *Psychological research*, pages 1–14, 2018.

R. Manduchi, A. Castano, A. Talukder, and L. Matthies. Obstacle detection and terrain classification for autonomous off-road navigation. *Autonomous robots*, 18(1):81–102, 2005.

C. Mead. Neuromorphic electronic systems. *Proceedings of the IEEE*, 78(10):1629–1636, 1990.

P. A. Merolla, J. V. Arthur, R. Alvarez-Icaza, A. S. Cassidy, J. Sawada, F. Akopyan, B. L. Jackson, N. Imam, C. Guo, Y. Nakamura, et al. A million spiking-neuron integrated circuit with a scalable communication network and interface. *Science*, 345(6197):668–673, 2014.

O. Michel. Cyberbotics ltd. webots™: professional mobile robot simulation. *International Journal of Advanced Robotic Systems*, 1(1):39–42, 2004.

K. Miyazaki, K. W. Miyazaki, A. Yamanaka, T. Tokuda, K. F. Tanaka, and K. Doya. Reward probability and timing uncertainty alter the effect of dorsal raphe serotonin neurons on patience. *Nature communications*, 9(1):1–11, 2018.

F. Mondada, M. Bonani, X. Raemy, J. Pugh, C. Cianci, A. Klaptocz, S. Magnenat, J. christophe Zufferey, D. Floreano, and A. Martinoli. The e-puck, a robot designed for education in engineering. In *Proceedings of the 9th Conference on Autonomous Robot Systems and Competitions*, pages 59–65, Castelo Branco, Portugal, 2009. IPCB.

S. Moradi, N. Qiao, F. Stefanini, and G. Indiveri. A scalable multicore architecture with heterogeneous memory structures for dynamic neuromorphic asynchronous processors (dynaps). *IEEE transactions on biomedical circuits and systems*, 12(1):106–122, 2017.

E. I. Moser, E. Kropff, and M.-B. Moser. Place cells, grid cells, and the brain's spatial representation system. *Annu. Rev. Neurosci.*, 31:69–89, 2008.

V. Nair and G. E. Hinton. Rectified linear units improve restricted boltzmann machines. In *Proceedings of the 27th international conference on machine learning (ICML-10)*, pages 807–814, 2010.

J. Naude, S. Tolu, M. Dongelmans, N. Torquet, S. Valverde, G. Rodriguez, S. Pons, U. Maskos, A. Mourot, F. Marti, and P. Faure. Nicotinic receptors in the ventral tegmental area promote uncertainty-seeking. *Nat Neurosci*, 19(3):471–478, 2016. ISSN 1546-1726 (Electronic) 1097-6256 (Linking). doi: 10.1038/nn.4223.

E. Neftci, H. Mostafa, and F. Zenke. Surrogate gradient learning in spiking neural networks. *Signal Processing Magazine, IEEE*, Dec 2019. (accepted).

E. O. Neftci, B. U. Pedroni, S. Joshi, M. Al-Shedivat, and G. Cauwenberghs. Stochastic synapses enable efficient brain-inspired learning machines. *Frontiers in neuroscience*, 10: 241, 2016.

E. O. Neftci, C. Augustine, S. Paul, and G. Detorakis. Event-driven random back-propagation: Enabling neuromorphic deep learning machines. *Frontiers in neuroscience*, 11:324, 2017.

S. Nolfi, J. Bongard, P. Husbands, and D. Floreano. Evolutionary robotics. In *Springer Handbook of Robotics*, pages 2035–2068. Springer, Cham, 2016. ISBN 978-3-319-32552-1.

J. O'Keefe. Place units in the hippocampus of the freely moving rat. *Experimental Neurology*, 51(1):78–109, 1976.

J. O'keefe and L. Nadel. *The hippocampus as a cognitive map*. Oxford: Clarendon Press, 1978.

J. M. Olson, K. Tongprasearth, and D. A. Nitz. Subiculum neurons map the current axis of travel. *Nature Neuroscience*, 20(2):170–172, 2017.

J. M. Olson, J. K. Li, S. E. Montgomery, and D. A. Nitz. Secondary motor cortex transforms spatial information into planned action during navigation. *Current Biology*, 30(10):1845–1854.e4, 2020. ISSN 0960-9822. doi: 10.1016/j.cub.2020.03.016.

J. M. Olson, A. B. Johnson, L. Chang, E. L. Tao, X. Wang, and D. A. Nitz. Complementary maps for location and environmental structure in ca1 and subiculum, 2021.

N. Oros and J. L. Krichmar. Smartphone based robotics: Powerful, flexible and inexpensive robots for hobbyists, educators, students and researchers. Technical Report 13-16, Center for Embedded Computer Systems, University of California, Irvine, Irvine, California, 2013.

N. Oros, A. A. Chiba, D. A. Nitz, and J. L. Krichmar. Learning to ignore: a modeling study of a decremental cholinergic pathway and its influence on attention and learning. *Learning & Memory*, 21(2):105–118, 2014.

D. Pacella, M. Ponticorvo, O. Gigliotta, and O. Miglino. Basic emotions and adaptation. a computational and evolutionary model. *PLOS ONE*, 12(11):1–20, 2017. doi: 10.1371/journal.pone.0187463.

A. Paszke, S. Gross, S. Chintala, G. Chanan, E. Yang, Z. DeVito, Z. Lin, A. Desmaison, L. Antiga, and A. Lerer. Automatic differentiation in PyTorch. In *NIPS Autodiff Workshop*, 2017.

D. Pathak, P. Agrawal, A. A. Efros, and T. Darrell. Curiosity-driven exploration by self-supervised prediction. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pages 16–17, 2017.

P. Pauls. A primer on the fundamental concepts of neuroevolution. `https://towardsdatascience.com/a-primer-on-the-fundamental-concepts-of-neuroevolution-9068f532f7f7`, 2020. Accessed: 2021-04-28.

F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

J. Pei, L. Deng, S. Song, M. Zhao, Y. Zhang, S. Wu, G. Wang, Z. Zou, Z. Wu, W. He, et al. Towards artificial general intelligence with hybrid tianjic chip architecture. *Nature*, 572 (7767):106–111, 2019.

O. Potvin, F. Y. Doré, and S. Goulet. Contributions of the dorsal hippocampus and the dorsal subiculum to processing of idiothetic information and spatial memory. *Neurobiology of Learning and Memory*, 87(4):669–678, 2007. doi: 10.1016/j.nlm.2007.01.002.

I. Rechenberg. Evolution strategy: Optimization of technical systems by means of biological evolution. *Fromman-Holzboog, Stuttgart*, 104:15–16, 1973.

F. Sargolini, M. Fyhn, T. Hafting, B. L. McNaughton, M. P. Witter, M.-B. Moser, and E. I. Moser. Conjunctive representation of position, direction, and velocity in entorhinal cortex. *Science*, 312(5774):758–762, 2006. doi: 10.1126/science.1125572.

M. Sarter, M. E. Hasselmo, J. P. Bruno, and B. Givens. Unraveling the attentional functions of cortical cholinergic inputs: interactions between signal-driven and cognitive modulation of signal detection. *Brain Research Reviews*, 48(1):98–111, 2005.

B. Schrauwen, D. Verstraeten, and J. Van Campenhout. An overview of reservoir computing: theory, applications and implementations. In *Proceedings of the 15th european symposium on artificial neural networks. p. 471-482 2007*, pages 471–482, 2007.

H.-P. Schwefel. Evolutionsstrategien für die numerische optimierung. In *Numerische Optimierung von Computer-Modellen mittels der Evolutionsstrategie*, pages 123–176. Springer, 1977.

H.-P. Schwefel and G. Rudolph. Contemporary evolution strategies. In *European conference on artificial life*, pages 891–907. Springer, 1995.

E. O. Scott and S. Luke. Ecj at 20: toward a general metaheuristics toolkit. In *Proceedings of the genetic and evolutionary computation conference companion*, pages 1391–1398, 2019.

R. R. Selvaraju, M. Cogswell, A. Das, R. Vedantam, D. Parikh, and D. Batra. Grad-cam: Visual explanations from deep networks via gradient-based localization. In *2017 IEEE International Conference on Computer Vision (ICCV)*, pages 618–626. IEEE, 2017.

P. E. Sharp and C. Green. Spatial correlates of firing patterns of single cells in the subiculum of the freely moving rat. *Journal of Neuroscience*, 14(4):2339–2356, 1994.

E. E. Smith and J. Jonides. Storage and executive processes in the frontal lobes. *Science*, 283(5408):1657–1661, 1999.

L. Sokoloff. The metabolism of the central nervous system in vivo. *Handbook of Physiology, section, I, Neurophysiology*, 3:1843–64, 1960.

K. O. Stanley and R. Miikkulainen. Evolving neural networks through augmenting topologies. *Evolutionary Computation*, 10(2):99–127, 2002.

K. O. Stanley, D. B. D'Ambrosio, and J. Gauci. A hypercube-based encoding for evolving large-scale neural networks. *Artificial Life*, 15(2):185–212, 2009. doi: 10.1162/artl.2009.15.2.15202.

Y. Sun, S. Jin, X. Lin, L. Chen, X. Qiao, L. Jiang, P. Zhou, K. G. Johnston, P. Golshani, Q. Nie, T. C. Holmes, D. A. Nitz, and X. Xu. Ca1-projecting subiculum neurons facilitate object–place learning. *Nature Neuroscience*, 22(11):1857–1870, 2019.

C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich. Going deeper with convolutions. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1–9, 2015.

G. Tanaka, T. Yamane, J. B. Héroux, R. Nakane, N. Kanazawa, S. Takeda, H. Numata, D. Nakano, and A. Hirose. Recent advances in physical reservoir computing: A review. *Neural Networks*, 115:100–123, 2019.

J. Tanner and L. Itti. Goal relevance as a quantitative model of human task relevance. *Psychological review*, 124(2):168, Mar 2017.

J. Tanner and L. Itti. A top-down saliency model with goal relevance. *Journal of vision*, 19 (1):1–16, Jan 2019.

J. S. Taube. The head direction signal: origins and sensory-motor integration. *Annu. Rev. Neurosci.*, 30:181–207, 2007.

J. S. Taube, R. U. Muller, and J. B. Ranck. Head-direction cells recorded from the postsubiculum in freely moving rats. i. description and quantitative analysis. *Journal of Neuroscience*, 10(2):420–435, 1990.

E. C. Tolman. Cognitive maps in rats and men. *Psychological review*, 55(4):189–208, 1948. doi: 10.1037/h0061626.

E. C. Tolman and C. H. Honzik. Introduction and removal of reward, and maze performance in rats. *University of California publications in psychology*, 1930.

J. K. Tsotsos, M. P. Eckstein, and M. S. Landy. Computational models of visual attention. *Vision Res*, 116(Pt B):93–94, 2015. ISSN 1878-5646 (Electronic) 0042-6989 (Linking). doi: 10.1016/j.visres.2015.09.007.

S. Vyas, M. D. Golub, D. Sussillo, and K. V. Shenoy. Computation through neural population dynamics. *Annual Review of Neuroscience*, 43(1):249–275, 2020.

K. Walas. Terrain classification and negotiation with a walking robot. *Journal of Intelligent & Robotic Systems*, 78(3-4):401–423, 2015.

J. X. Wang, Z. Kurth-Nelson, D. Kumaran, D. Tirumala, H. Soyer, J. Z. Leibo, D. Hassabis, and M. Botvinick. Prefrontal cortex as a meta-reinforcement learning system. *Nature neuroscience*, 21(6):860–868, 2018.

Webots. Webots user guide r2020a – gctronic' e-puck. `https://cyberbotics.com/doc/guide/epuck#gctronic-e-puck`, 2020. Accessed: 2020-05-04.

S. M. Weisberg and N. S. Newcombe. How do (some) people make a cognitive map? routes, places, and working memory. *Journal of Experimental Psychology: Learning, Memory, and Cognition*, 42(5):768, 2016.

S. M. Weisberg and N. S. Newcombe. Cognitive maps: some people make them, some people struggle. *Current directions in psychological science*, 27(4):220–226, 2018.

J. S. Weszka, C. R. Dyer, and A. Rosenfeld. A comparative study of texture measures for terrain classification. *IEEE transactions on Systems, Man, and Cybernetics*, SMC-6(4): 269–285, 1976.

S. I. Wiener and J. S. Taube. *Head direction cells and the neural mechanisms of spatial orientation*. MIT Press, 2005.

E. Wieser and G. Cheng. Eo-mtrnn: evolutionary optimization of hyperparameters for a neuro-inspired computational model of spatiotemporal learning. *Biological Cybernetics*, 114(3):363—-387, 2020.

M. Wilson and B. McNaughton. Dynamics of the hippocampal ensemble code for space. *Science*, 261(5124):1055–1058, 1993a. ISSN 0036-8075. doi: 10.1126/science.8351520.

M. A. Wilson and B. L. McNaughton. Dynamics of the hippocampal ensemble code for space. *Science*, 261(5124):1055–1058, 1993b.

R. G. Winther. Darwin on variation and heredity. *Journal of the History of Biology*, 33(3): 425–455, 2000.

R. A. Wise. Dopamine, learning and motivation. *Nature reviews neuroscience*, 5(6):483–494, 2004.

D. R. Wozny, U. R. Beierholm, and L. Shams. Probability matching as a computational strategy used in perception. *PLoS Comput Biol*, 6(8), 2010. ISSN 1553-7358 (Electronic) 1553-734X (Linking). doi: 10.1371/journal.pcbi.1000871.

J. Xing, X. Zou, and J. Krichmar. Neuromodulated patience for robot and self-driving vehicle navigation. In *IEEE International Joint Conference on Neural Networks (IJCNN)*, 2020.

T. Yamamoto, T. Nishino, H. Kajima, M. Ohta, and K. Ikeda. Human support robot (hsr). In *ACM SIGGRAPH 2018 Emerging Technologies*, page 11. ACM, 2018.

G. R. Yang, M. R. Joglekar, H. F. Song, W. T. Newsome, and X.-J. Wang. Task representations in neural networks trained to perform many cognitive tasks. *Nature neuroscience*, 22(2):297–306, 2019.

M. Yang, S.-C. Liu, and T. Delbruck. A dynamic vision sensor with 1% temporal contrast sensitivity and in-pixel asynchronous delta modulator for event encoding. *IEEE Journal of Solid-State Circuits*, 50(9):2149–2160, 2015.

A. J. Yu and P. Dayan. Uncertainty, neuromodulation, and attention. *Neuron*, 46(4):681–692, 2005. ISSN 0896-6273 (Print) 0896-6273 (Linking). doi: 10.1016/j.neuron.2005.04.026.

L. Zaborszky. The modular organization of brain systems. basal forebrain: the last frontier. In *Progress in brain research*, volume 136, pages 359–372. Elsevier, 2002.

F. Zenke and S. Ganguli. Superspike: Supervised learning in multilayer spiking neural networks. *Neural computation*, 30(6):1514–1541, 2018.

J. Zhang, S. A. Bargal, Z. Lin, J. Brandt, X. Shen, and S. Sclaroff. Top-down neural attention by excitation backprop. *International Journal of Computer Vision*, 126(10): 1084–1102, 2018.

B. Zhou, A. Khosla, A. Lapedriza, A. Oliva, and A. Torralba. Learning deep features for discriminative localization. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2921–2929, 2016.

X. Zou, T. Hwu, J. Krichmar, and E. Neftci. Terrain classification with a reservoir-based network of spiking neurons. In *2020 IEEE International Symposium on Circuits and Systems (ISCAS)*, pages 1–5, Sevilla, Spain, 2020a. IEEE. doi: 10.1109/ISCAS45731.2020.9180740.

X. Zou, S. Kolouri, P. K. Pilly, and J. L. Krichmar. Neuromodulated attention and goal-driven perception in uncertain domains. *Neural Networks*, 125:56–69, 2020b. doi: 10.1016/j.neunet.2020.01.031.

X. Zou, E. O. Scott, A. B. Johnson, K. Chen, D. A. Nitz, K. A. De Jong, and J. L. Krichmar. Neuroevolution of a recurrent neural network for spatial and working memory in a simulated robotic environment. *arXiv preprint arXiv:2102.12638*, 2021.